EASY BOOK IN FRAMEWORK

PART 2

EASY BOOK IN FRAMEWORK



Authorship: Eng / Malek

Malek



Collection And Generics

مقدمة

بسم الله الرحمن الرحيم وبه نستعين أحبتى فى الله

ان شاء الله سيكون هذا الكتاب الأصدار الأخير من الفريم ورك وبهذا نكون قد الممنا بالفريم ورك ولكن أريد منكم ان تقرأو الدروس بكل وضوح وبكل تركيز ودقه وأرجو ان تقومو بالتطبيق المباشر على كل مثال نقوم بتطبيقه ولا تنسخ الأكواد وتلصقها لأنك لو فعلت هذا لم تستفاد أبدا .. لابد ان تكتب الاكواد بيدك لتأخذ على مود الاكواد وتستطيع تطبيق أمثله وحدك بدون معلم . كل ماعليك ان شاء الله ان تقوم بقراءه الأمثله بدقه وتركيز وتقوم بالتطبيق المباشر فى نفس الوقت ولم تؤجل التطبيق لوقت أخر.

اود أيضا ان الفت نظركم الى ماهيه الفريم ورك بكل وضوح واختصار
بعض الطلبةاو القارئين للجزء الأول من هذا الكتاب كان يراسلنا على الايميل وكانو يظنون ان هذا الكتاب يشرح لغه السى شارب ... ولكن هذا ليس بصحيح نحن لم نقم بشرح السى شارب ولكننا نقوم بشرح الفريم ورك عموما ... باختصار ان مانشرحه هذا نستطيع تطبيه على الفيجوال نت والسى شارب نت والسى بلاس نت.... الخ
(نستطيع تطبيقه على كل اللغات الدوت نت) ولكن تختلف طريقه كتابه الأكواد فى كل لغه عن الأخرى أى تطبيق هذه الأكواد فى بيئه السى الشارب مثلا يختلف عن طريقه تطبيقها فى الفيجوال يختلف عن طريقه كتابتها فى السى بلاس لأن كل لغه تختص بطريقه كتابه للأكواد معينه فلايجوز ان نطبق بنفس الطريقه (اعتقد ان الكلام واضح ولا يشوبه شىء من الغرابه) ومانقوم بتطبيقه فى هذا الجزء من الكتاب والجزء الأول كان تحت بيئه الفيجوال نت
ايضا اود ان انوه على أنه بعد هذا الاصدار ستتوالى الاعمال ان شاء الله وسنقوم باصدار كتاب أخر يشرح كيفيه البرمجه على الطريقه الحديثه Object Oriented Programing(OOP)
ويتبع هذا الشرح مشروع البنك على طريقه الـ OOP
ادعو لنا بالتوفيق .

ملحوظه!!

هذه الأصدارات من الكتب مجانيه لوجه الله تعالى وغير مسموح ببيعها تحت اى ظرف من الظروف فهى لتعليم من لايملك حق التعليم او الكورسات .

اذا استفدت من هذه الاصدارات قم بالدعاء لى ولوالدى

الوحده الاولى :-

ARRAY LIST

هى نوع من انواع المصفوفات الكوليكشن تقبل ان تحمل جميع انواع الداتا سواء كانت تاريخ او رقميه او حرفيه وتختلف عن المصفوفه العاديه بعض اختلافات سوف تلاحظها عند شرح الأكواد ... لاحظ كيفيه استخدامها.

```
Imports System.Collections
Module Module1

    Sub Main()

        Dim Array As New ArrayList
        Array.Add("Hello")
        Array.Add("Im Captian Malek")
        Array.Add("Im 22 Years Old")
        Array.Add(100)
        Array.Add(200)
        Array.Add(Array(3) + Array(4))


        Console.WriteLine(Array(0))
        Console.WriteLine(Array(1))
        Console.WriteLine(Array(2))
        Console.WriteLine(Array(5))
        Console.WriteLine("Sum {0} +  {1}= {2}", Array(3), Array(4), 
Array(5))
        Console.ReadKey()

    End Sub
End Module
```


الشرح

اولا قمنا باستيراد الـ Collection قبل الموديول وذلك لاتاحه استخدام الـ Array List لأنها من انواع الـ Collection

ثم قمنا بأستخدامها تحت الـ Sub Main()

```
Dim Array As New ArrayList
```


اى قم بحجز متغير بألاسم Array يكون من النوع ArrayList

ثم قمنا بأضافه بعض البيانات داخل المصفوفه ArrayList ... بيانات من النوع النصيه مثل

Hello , Im Captian Malek , Im 22 Years Old

وبعضها الأخر رقميه مثل 100 ,200

```
Array.Add("Hello")
Array.Add("Im Captian Malek")
Array.Add("Im 22 Years Old")
Array.Add(100)
Array.Add(200)
Array.Add(Array(3) + Array(4))
```

فكما قولنا ان هذه المصفوفه تقبل اى نوع من انواع البيانات فهى من النوع Object الذى يقبل جميع انواع البيانات

اذا فما هو معنى هذا الكود ؟

```
Array.Add(Array(3) + Array(4))
```

ركز معى لحظات لتفهم ماهو معنى هذا الكود

لقد سبق لنا ان وضحنا المصفوفات فى الجزء الأول من الكتاب وقولنا ان المصفوفه غالبا ماتبدأ بالاندكس صفر .... فاذا كانت المصفوفه تتكون من اربع اجزاء .. اذا فهى تنتهى عند الرقم 3

بمعنى اوضح ان كانت المصفوفه تتكون من 4 اجزاء اذا

الجزء الأول يكون رقمه (صفر) و الجزء الثانى يكون رقمه (1) والجزء الثالث يكون رقمه(2) والجزء الرابع يكون رقمه (3)

والأن نأتى الى شرح هذا الكود `Array.Add(Array(3) + Array(4))`

نحن فى بدايه الكود قمنا بأضافه بعض البيانات داخل المصفوفه اذا فكم هى عدد البيانات التى قمنا باضافتها ؟

١ - Hello ... ويكون رقم الاندكس له صفر كما اتفقنا
٢ - Im Captian Malek ... ويكون رقم الاندكس له 1
٣ - Im 22 Years Old ... ويكون رقم الاندكس له 2
٤ - 100 ... ويكون رقم الاندكس له 3
٥ - 200 ... ويكون رقم الاندكس له 4
٦ - `Array(3) + Array(4)` ... ويكون رقم الاندكس له 5

ونحن فى الكود أمرناه بجمع الاندكس 3 مع الاندكس 4 ... ليكون الناتج هو 100+200 = 300 وهذا يؤكد لك انها تتعامل مع النوع الرقمى من البيانات

فان كانت لاتقبل الانواع الرقميه كانت عرضت الرقم هكذا 100200

فان امرنا الكونسول بطباعه النتائج هكذا

```
Console.WriteLine(Array(0))
Console.WriteLine(Array(1))
Console.WriteLine(Array(2))
Console.WriteLine(Array(5))
Console.WriteLine("Sum {0} +  {1}= {2}", Array(3), Array(4), Array(5))
Console.ReadKey()
```

ولن أقوم بشرح هذا الكود ... لانه من خصائص الكونسول وتم شرحه فى الجزء الأول من الكتاب

وهذه هى النتيجه



بعض خصائص الـ ArrayList

الخاصيه الاولى :- هى التعديل على عنصر مضاف من قبل بدلاله رقم الاندكس له

```
Array.Insert(3, "Malek")
```

فهنا امرناه بتعديل مافى الاندكس رقم 3 .. ان يحذف مافيه ويضع مكانه Malek
فان قمنا بتطبيقه على المثال السابق نجد انه سيحذف مافى الاندكس 3 (100) ويضع مكانه Malek

الخاصيه الثانيه :- الحذف بدلاله البيان

```
Array.Remove("Hello")
```

هنا امرناه بالحذف بدلاله البيان داخل المصفوفه ... فعند تطبيقه على المثال السابق يقوم بالبحث داخل المصفوفه حتى يجد البيان Hello
وفى المثال السابق كان هذا البيان فى اول جزء من المصفوفه .. الذى هو رقمه صفر فعند تطبيقه يزيل مافى الاندكس صفر لتصبح ترتيبه الاندكس هى كالأتى :-

١ - Im Captian Malek ... ويكون رقم الاندكس له صفر
٢ - Im 22 Years Old ... ويكون رقم الاندكس له 1
٣ - 100 ... ويكون رقم الاندكس له 2
٤ - 200 ... ويكون رقم الاندكس له 3
٥ - Array(3) + Array(4) ... ويكون رقم الاندكس له 4

الخاصيه الثالثه :- الحذف بدلاله رقم الاندكس

```
Array.RemoveAt(4)
```

هنا امرناه بحذف مافى الاندكس رقم 4
وبتطبيقه على المثال السابق يقوم بحذف مافى الاندكس رقم 4 وهو (200) لتصبح المصفوفه هكذا

١ - Hello ... ويكون رقم الاندكس له صفر
٢ - Im Captian Malek ... ويكون رقم الاندكس له 1
٣ - Im 22 Years Old ... ويكون رقم الاندكس له 2
٤ - 100 ... ويكون رقم الاندكس له 3
٥ - Array(3) + Array(4) ... ويكون رقم الاندكس له 4

الخاصيه الرابعه :- حذف اكتر من جزء من المصفوفه بدلاله الاندكس

```
Array.RemoveRange(3, 2)
```

هنا امرناه بحذف 2 جزء من المصفوفه ابتداءا من الاندكس رقم 3
اى سيقوم بحذف الاندكس رقم 4 والاندكس رقم 5
فان قمنا بتطبيقه على المثال السابق اصبحت المصفوفه هكذا

١ - Hello ... ويكون رقم الاندكس له صفر
٢ - Im Captian Malek ... ويكون رقم الاندكس له 1
٣ - Im 22 Years Old ... ويكون رقم الاندكس له 2
٤ - Array(3) + Array(4) ... ويكون رقم الاندكس له 3

البحث داخل الـ Array List بدلاله البيان

والأن سنقوم بتطبيق مثال جديد يوضح كيفيه البحث داخل الأراى ليست بدلاله البيان الذى بداخلها ... ضع تركيزك فى هذا المثال لتفهمه بطريقه صحيحه.

```
Imports System.Collections
Module module1
    Sub main()

        Dim Array3 As New ArrayList

        Array3.Add("AAA")
        Array3.Add("BBB")
        Array3.Add("CCC")

        Console.Write("Type The Name You Want To Search: ")
        Dim H As String = Console.ReadLine
        If Array3.Contains(H) Then
            Dim T = Array3.LastIndexOf(H)
            Console.WriteLine("Index Number{0}", T)
            Console.ReadKey()

        End If

        Console.WriteLine("Sorry Name Not Found")
        Console.ReadKey()

    End Sub

End Module
```


الشرح

فى المثال السابق قمنا باستيراد دوال الـ Collections لنستطيع استعمال الاراى ليست كما شرحنا من قبل

```
Imports System.Collections
```

ثم قمنا بحجز متغير بالأسم Array3 من نوع المصفوفه اراى ليست
ثم قمنا باضافه بعض البيانات داخل هذه المصفوفهلكى يتم البحث عنها

```
Dim Array3 As New ArrayList

Array3.Add("AAA")
Array3.Add("BBB")
Array3.Add("CCC")
```

ثم امرنا الكونسول ان يطلب من المستخدم ان يدخل اسم البيان المراد البحث عنه داخل المصفوفه ... وان يضع قيمه ما ادخله المستخدم فى المتغير H

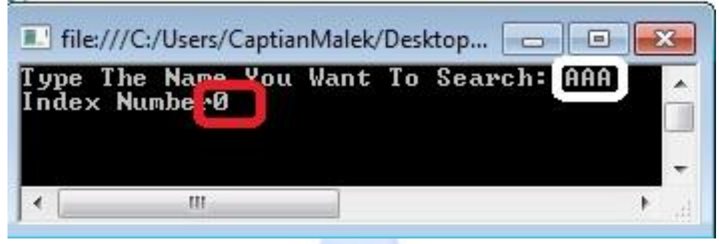
```
Console.Write("Type The Name You Want To Search: ")
Dim H As String = Console.ReadLine
```

ثم قمنا بوضع جمله شرط ... تقوم بالبحث داخل المصفوفه عن القيمه التى ادخلها المستخدم فان وجدتها تذهب فى الطريق الحقيقى وهو طباعه نتائج البحث ...
وان لم تجدها ذهبت فى الطريق الغير حقيقى .. الذى بعد كلمه End If

```
If Array3.Contains(H) Then
Dim T = Array3.LastIndexOf(H)
Console.WriteLine("Index Number{0}", T)
Console.ReadKey()
End If

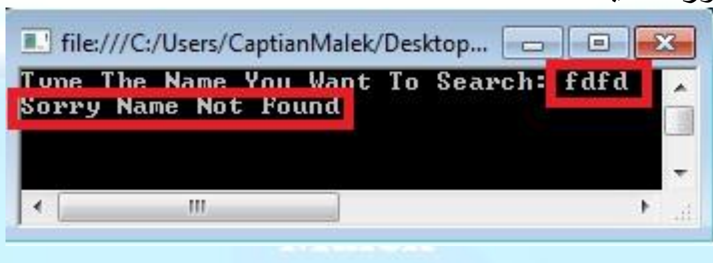
Console.WriteLine("Sorry Name Not Found")
Console.ReadKey()
```


ركز فى الكود كويس ... جمله الشرط بتقول ان كانت المصفوفه تحتوى على مافى المتغير H وكما قولنا ان هذا المتغير الان يحمل ما قام بادخاله المستخدم
فان وجدت هذه القيمه فى المصفوفه ...اذا من فضلك اعرض لى رقم الاندكس الخاص بها فى هذه المصفوفه
فان قام المستخدم بادخال AAA
فتكون النتيجه هى الاندكس صفر
كماترى فى الصوره التاليه ... هذه هى نتيجه البحث



وان قام المستخدم بادخال اى قيمه اخرى غير موجوده فى المصفوفه
فتكون النتيجه

كما ترى فى الصوره التاليه



استخدام الـ Array List مع المصفوفه العاديه

سوف نتعرض الأن لكيفيه استخدام المصفوفه العاديه مع المصفوفه اراى ليست
وسوف نتعرض لكيفيه ارسال كل قيم المصفوفه العاديه الى المصفوفه اراى ليست وكيفيه ارسال قيمه واحده من المصفوفه العاديه الى المصفوفه اراى ليست

وهذا مثال يجمع ماقولنا به:-

```
Imports System.Collections
Module module1
    Sub main()

        Dim Array2 As New ArrayList

        Dim M(2) As String
        M(0) = "A"
        M(1) = "B"
        M(2) = "C"

        Array2.AddRange(M)
        Array2.Add(M(1))
        Array2.InsertRange(1, M)



        Dim I As Integer = Array2.Count
        Dim s As Integer
        For s = 0 To I - 1
            Console.WriteLine(Array2(s))
        Next
        Console.ReadKey()

    End Sub
End Module
```


الشرح

فى المثال السابق قمنا باستيراد دوال الـ Collections لنستطيع استعمال الاراى ليست كما شرحنا من قبل عن طريق الكود
Imports System.Collections

ثم قمنا بحجز متغير بالأسم Array2 من نوع المصفوفه اراى ليست
ولم نقم باضافه اى اجزاء داخل هذه المصفوفه ... لاننا سوف نضع بها البيانات عن طريق ارسالها من مصفوفه عاديه كما سترى فيما بعد ... تابع معى

```
Dim Array3 As New ArrayList
```

ثم قمنا بحجز مصفوفه عاديه من النوع النصى ووضعنا باجزائها بيانات نصيه عن طريق الكود

```
Dim M(2) As String
M(0) = "A"
M(1) = "B"
M(2) = "C"
```

والأن تبقى ان نرسل قيم هذه المصفوفه العاديه الى المصفوفه اراى ليست .. فكيف يتم ذلك ؟
الجواب :- يتم ذلك عن طريق الخاصيه Add Range
هذه الخاصيه هى من احد خواص الاراى ليست ... وتقوم هذه الخاصيه باضافه العناصر المحدده الى الاراى ليست . ويتم ذ لك عن طريق الكود

```
Array2.AddRange(M)
```

فهذا الكود يعنى .. اى قم باضافه كل محتويات المصفوفه العاديه M الى المصفوفه اراى ليست
وتنفيذا لهذا الكود تصبح المصفوفه اراى ليست تحتوى على الاجزاء

A
B
C

وايضا يمكننا اضافه عنصر واحد فقط من عناصر المصفوفه العاديه وليس اضافه كل عناصرها . وذلك يتم عن طريق الخاصيه Add
هذه الخاصيه ايضا من احد خواص الأراى ليست ولكن تقوم باضافه عنصر واحد بدلاله الاندكس له ويتم ذلك عن طريق الكود

```
Array2.Add(M(1))
```

فهذا الكود يعنى .. اى قم باضافه محتوى المصفوفه العاديه M الى المصفوفه اراى ليست ولكن قم باضافه المحتوى الذى يحمل الاندكس رقم 1 فقط .
فان قمنا بتنفيذ هذا الكود مع الكود السابق .. يضيف على محتويات المصفوفه قيمه اخرى هى قيمه الاندكس رقم واحد .. اى سيقوم باضاف العنصر **B**
لتصبح المصفوفه اراى ليست تحتوى على الاجزاء

A
B
C
B

وايضا يمكننا اضافه مجموعه عناصر محدده فنقوم بأمر المصفوفه اراى ليست ان تأخذ قيم من المصفوفه العاديه نقوم بتحديدها وذلك يتم عن طريق الخاصيه Insert Range
وهذه الخاصيه من احد خواص الاراى ليست .. التى تقوم باضافه قيم محدد الى الاراى ليست ويتم ذلك عن طريق الكود

```
Array2.InsertRange(1, M)
```

ومعنى هذا الكود .. اى قم باضافه عناصر المصفوفه العاديه الى المصفوفه اراى ليست

ولكن بدأً من العنصر صاحب الاندكس رقم 1 حتى نهايه المصفوفه... بمعنى ادق انه سوف يقوم باضافه عناصر المصفوفه العاديه كلها الى المصفوفه اراى ليست عدى العنصر المحتوى على الاندكس صفر .
وتطبيقا لهذا الكود يضيف على محتويات المصفوفه اراى ليست العناصر B , C
فان قمنا بتطبيقه بعد الاكواد السابقه اصبحت محتويات المصفوفه اراى ليست

A
B
C
B
B
C

والأن سنقوم بعمل دواره لعرض كل محتويات الأراى ليست على شاشه الكونسول
وتكون فكره هذه الدواره قائمه على .. حجز متغير يحمل عدد اجزاء المصفوفه اراى ليست
ثم حجز تغير اخر نستخدمه كمعامل فى الدواره ليتم عن طريقه عرض النتائج
ثم يكون نص الدواره .. قم بعرض محتوى المعامل على شاشه الكونسول حتى يصل الى نهايه جزء فى المصفوفه
وهذا هو كود الدواره .. سوف يهيء لك الفكره بطريقه افضل

```
Dim I As Integer = Array2.Count
Dim s As Integer
For s = 0 To I - 1
Console.WriteLine(Array2(s))
Next
Console.ReadKey()
```

فى هذا الكود قمنا بحجز متغير .. ووضعنا به عدد اجزاء المصفوفه اراى ليست
ثم قمنا بحجز متغير اخر ليكون معامل ليتم عن طريقه عرض محتويات المصفوفه اراى ليست
ثم قمنا بكتابه نص الدواره .. الذى ينص على . افعل عندما يكون المعامل يبدأ من الرقم صفر حتى يصل الى نهايه عدد اجزاء المصفوفه ناقص واحد (- 1)
قم بعرض المحتوى الذى رقم الاندكس له هو رقم المعامل
والمعامل دلوقتى جوه الدواره كل لفه رقمه بيزيد واحد ... وهو يبدأ من الرقم صفر وينتهى حتى عدد اجزاء المصفوفه

والسؤال هنا : لماذا قولنا حتى نهايه عدد اجزاء المصفوفه ناقص واحد (- 1) ؟
اعتقد ان اجابه هذا السؤال سوف تغير الفكره فى عقلك .. وتوضح كيفيه الأداء التكنيكي للدواره فقم بوضع كل تركيزك معى لتفهم ماهو الاسلوب التكنيكي لهذه الدواره .
بدايه فكر معى الان كم هو عدد اجزاء المصفوفه اراى ليست الان ؟ المصفوفه الان تحتوى على 6 اجزاء وهما كالتالى :-

A 1
B 2
C 3
B 4
B 5
C 6

وما هو كان شرطنا فى جمله الدواره ؟ ... شرطنا فى جمله الدواره ان يبدأ المعامل العد من الصفر حتى نهايه المصفوفه ناقص واحد . تخيل معى انه ينتهى عند نهايه اجزاء المصفوفه فقط فكما قولنا ان المصفوفه هى 6 اجزاء . اذن سوف ينتهى العد عند رقم 6

صفر >> 1 >> 2 >> 3 >> 4 >> 5 >> 6

اذا سيقوم بعمل 7 لفات بدايه من اللفه صفر حتى اللفه 6
فى كل لفه سيقوم بعرض صنف بدلاله الاندكس الذى رقمه هو رقم هذه اللفه فتخيل معى كل لفه الأن

صفر >> 1 >> 2 >> **3** >> 4 >> 5 >> 6
A >> B >> C >> **B** >> B >> C >> ??

انتهت الوحده الأولى :
كن على يقين عزيزى القارىء انه لطالما قمت بقراءه الوحده الأولى بتركيز وفهمت محتوياتها انك ستفهم باقى الوحدات الأتيه بكل سهوله .. لأن كل الوحدات الأتيه سوف تترتب على ماقبلها



الوحده الثانيه :-

QUEUE

وهو نوع من انواع المصفوفات الكوليكشن .. نقوم باستخدامه فى اضافه بعض البيانات مثل الارای ليست ثم نقوم بعرضها فى اى وقت نريده ... ولكن يختلف عن الارای ليست فى انه يعمل بمبدأ الداخل اولا يخرج اولا
Frist In Frist Out وتختصر هذه الجمله فبعض المطورين يطلقون عليها Fifo
اى ان البيان الذى دخل فى الأول يخرج فى الأول ويكون له استعمالات كثيره ومهمه جدا ولا غنى عنه فى المشاريع حيث يمكن استخدامه فى سوفت وير المطاعم او اى نظام يعمل بالطابور حيث يتم اضافه البيان بداخله ويقوم باخراجه بالطابور على حسب الداخل اولا .
وسوف نقوم بعرض مثال عليه لتصل لنا الفكره بصوره اوضح
تابع معى ...

```
Imports System.Collections
Module module1
    Sub main()

        Dim MQ As New Queue
        MQ.Enqueue("Ahmed")
        MQ.Enqueue("Mohamed")
        MQ.Enqueue("Abdo")
        Console.WriteLine(MQ.Dequeue)
        Console.ReadKey()


        Dim I As Integer
        For I = 0 To MQ.Count - 1
            Console.WriteLine(MQ.Dequeue)
        Next
        Console.ReadKey()

    End Sub
End Module
```


الشرح

نعتبر ان المثال السابق هو مجموعه من الافراد واقفين فى طابور فى مطعم معين لشراء الأكل
اولا : قمنا باستيراد دوال الـ Collections لنستطيع استعمال الكيو
عن طريق الكود

```
Imports System.Collections
```


ثم قمنا بحجز متغير بالاسم MQ من نوع المصفوفه كيو
ثم اضفنا بداخل المصفوفه مجموعه الافراد الواقفه فى الطابور .

وذلك عن طريق الخاصيه Enqueue التى تقوم باضافه البيانات داخل المصفوفه كيو
وذلك عن طريق الكود التالى :-

```
Dim MQ As New Queue
MQ.Enqueue("Ahmed")
MQ.Enqueue("Mohamed")
MQ.Enqueue("Abdo")
```

فان اردنا ان نأمر البرنامج باستخراج اول اسم قام بالحجز فى الطابور فيكون ذلك عن طريق الخاصيه Dequeue
هذه الخاصيه تقوم بعرض اول بيان دخل فى المصفوفه فان اردنا ان نقوم بعمل ذلك عن طريق الكود كان الكود كالأتى :-

```
Console.WriteLine(MQ.Dequeue)
```

ومعناه اننا امرانا الكونسول بكتابه اول اسم قمنا بتسجيله فى المصفوفه كيو.

-ماذا لو اردنا ان نقوم بعمل دواره تقوم بعرض كل المسجلين بالطابور على الترتيب بعرض الاول ثم الذى يليه حتى نهايه الطابور ؟
فكره الدواره تكون نفس فكره الدواره التى قمنا بشرحها فى اخر مثال فى الوحده الأولى لاتختلف نهائيا . فهى تكون عن طريق الكود الاتى :-


```
Dim I As Integer
For I = 0 To MQ.Count - 1
Console.WriteLine(MQ.Dequeue)
Next
Console.ReadKey()
```

هذه الدواره مضمونها باختصار
قمنا بحجز متغير من النوع العددى . ثم قمنا بعمل الدواره على الاساس الاتى
قم بالعرض الدائرى قيم البيانات الموجوده داخل المصفوفه كيو. بدايه من القيمه صفر للمتغير حتى نهايه المصفوفه ناقص واحد .
فلماذا قولنا حتى نهايه المصفوفه ناقص واحد ؟ قمنا بالاجابه على ذلك السؤال فى الدواره السابقه فى الوحده الاولى.

انتهت الوحده الثانيه:

الوحده الثالثه :-

STACK

وهو نوع من انواع المصفوفات الكوليكشن .. نقوم باستخدامه فى اضافه بعض البيانات ثم نقوم بعرضها فى اى وقت نريده ... وهو يعمل بمبدأ الداخل اخيرا يخرج اولا
Last In Frist Out وتختصرهذه الجمله فبعض المطورين يطلقون عليها Lifo
اى ان البيان الذى دخل فى الاخر يخرج فى الأول فهو يقوم بعكس عمل المصفوفه كيو .
ويكون له بعض الخواص التى تختلف عن خواص الكيو

فهو لديه خاصيه الـ**Push** التى تقوم باضافه العناصر داخله

ولديه ايضا الخاصيه **Pop** التى تقوم بعرض اخر صنف مسجل فى المصفوفه ستيك
وهذه الخاصيه تقوم بعرض اخر بيان او صنف مسجل فى المصفوفه ثم تقوم بنقل المؤشر على الصنف الذى يكون عليه الدور فى العرض.

ولديه ايضا الخاصيه**Peek** التى تقوم بعضر الصنف الذى يكون عليه الدور للعرض
او بمعنى اوضح تقوم بعرض الصنف الذى يقف عليه المؤشر

وسوف نقوم بعرض مثال عليه لتصل لنا الفكره بصوره اوضح
تابع معى ...

```
Imports System.Collections
Module Module1

    Sub Main()

        Dim Ms As New Stack
        Ms.Push("Nour")
        Ms.Push("Malek")
        Ms.Push("Shafee")
        Ms.Push("Salma")

        Console.WriteLine(Ms.Pop)
        Console.WriteLine(Ms.Peek)
        Console.ReadKey()


        Dim S = Ms.Count - 1
        Dim i As Integer
        For i = 0 To S
            Console.WriteLine(Ms.Pop)

        Next
        Console.ReadKey()

    End Sub

End Module
```

الشرح

اولا : قمنا باستيراد دوال الـ Collections لنستطيع استعمال الكيو عن طريق الكود

```
Imports System.Collections
```

ثم قمنا بحجز متغير بالاسم Ms من نوع المصفوفه ستيك
ثم اضفنا بداخل المصفوفه مجموعه بيانات.
وذلك عن طريق الخاصيه Push التى تقوم باضافه البيانات داخل المصفوفه ستيك
وذلك عن طريق الكود التالى :-

```
Dim Ms As New Stack
Ms.Push("Nour")
Ms.Push("Malek")
Ms.Push("Shafee")
Ms.Push("Salma")
```


فان اردنا ان نأمر البرنامج باستخراج اخر اسم قمنا بتسجيله فيكون ذلك عن طريق الخاصيه Pop
هذه الخاصيه تقوم بعرض اخر بيان دخل فى المصفوفه كما شرحنا مسبقا فان اردنا ان نقوم بعمل ذلك عن طريق الكود كان الكود كالأتى :-

```
Console.WriteLine(Ms.Pop)
```

ومعناه اننا امرانا الكونسول بكتابه اخر اسم قمنا بتسجيله فى المصفوفه ستيك.
وان اردنا ان نأمر البرنامج بعرض الاسم الذى سيكون عليه الدور فى العرض القادم او الاسم الذى يكون المؤشر واقفا عليه فيكون عن طريق الكود التالى :-

```
Console.WriteLine(Ms.Peek)
```

ومعناه اننا امرنا الكونسول بكتابه الاسم الذى سيكون عليه الدور فى المره القادمه

-فماذا لو اردنا ان نقوم بعمل دواره تقوم بعرض كل المسجلين بالمصفوفه على ترتيبها الخاص اى بعرض الاخير ثم مافوقه حتى بدايه المصفوفه؟
فكره الدواره تكون نفس فكره الدواره التى قمنا بشرحها سابقا
فهى تكون كالاتى:-


```
Dim i As Integer
For i = 0 To Ms.Count - 1
Console.WriteLine(Ms.Pop)
Next
Console.ReadKey()
```

هذه الدواره مضمونها باختصار
قمنا بحجز متغير من النوع العددى . ثم قمنا بعمل الدواره على الاساس الاتى
قم بالعرض الدائرى لقيم البيانات الموجوده داخل المصفوفه ستيك. بدايه من القيمه صفر للمتغير حتى نهايه المصفوفه ناقص واحد .
فلماذا قولنا حتى نهايه المصفوفه ناقص واحد ؟ قمنا بالاجابه على ذلك السؤال فى الدواره فى نهايه الوحده الاولى.

انتهت الوحده الثالثة :



الوحده الرابعه :-

HASH TABLE

هو ايضا من انواع المصفوفات الجينيرك ولكن له استخدام مختلف عن المصفوفات السابقه فهو يستخدم لتسجيل بيانات معينه ووضع مفتاح لكل بيان بحيث عند البحث داخله يمكننا البحث بهذا المفتاح ومن الممكن ان نضع المفتاح لكل بيان بأى نوع من انواع الداتا فلا يشترط ان يكون رقم فمن الممكن ان نضعه نص او اى نوع اخر من انواع البيانات حيث ان المفتاح من النوع اوبجكت .

بعض الخصائص او الميثود الخاصه بالهاش تيبل

الخاصيه Add

هذه الخاصيه تمكننا من اضافه بيان داخل المصفوفه هاش تيبل

الخاصيه Remove()

تمكننا هذه الخاصيه من مسح بيان معين داخل اجزاء المصفوفه هاش تيبل ويكون المسح بدلاله المفتاح حيث نقوم بوضع رقم المفتاح بين القوسين ان كان رقمى .. وان كان نصيا نقوم بوضعه بين القوسين وبين علامتي تنصيص فيكون هكذا Remove("MyKey")

خاصيه Values

لعرض قيمه معينه داخل اجزاء المصفوفه هاش تيبل

خاصيه Keys

لعرض مفتاح خاص بقيمه معينه داخل اجزاء المصفوفه هاش تيبل

خاصيه ContainsValue

تمكننا فى البحث داخل المصفوفه هاش تيبل عن قيمه بيان معين بداخل اجزائها

خاصيه ContainsKey

تمكننا فى البحث داخل المصفوفه هاش تيبل عن مفتاح خاص بقيمه معينه بداخل اجزائها

خاصيه Count

تمكننا بعرض عدد البيانات التى بداخل المصفوفه هاش تيبل

والأن سوف نتعرض الى مثال يوضح عمل الخواص او الميثود الخاصه بالهاش تيبل تابع معى ...

```
Imports System.Collections
Module module1
    Sub Main()

        Dim HT As New Hashtable

        HT.Add(1, "Mouse")
        HT(2) = "Monitor"
        HT(2) = "Sound"


        HT.Add("M", "Keyboard")
        Console.WriteLine(HT(1))
        Console.WriteLine(HT("M"))
        HT.Remove(1)  'مسح صنف بدلاله رقم المفتاح

       'لعرض عدد الاصناف المسجله بداخل الهاش تيبل
        Console.WriteLine("The item Count {0}", HT.Count)

        'لعرض القيم الخاصه بالمفاتيح داخل الهاش تيبل
        Console.WriteLine("================================")
        Dim K As Object
        For Each K In HT.Values
            Console.WriteLine(K)
        Next
        Console.ReadKey()

       'لعرض المفاتيح التى بداخل الهاش تيبل بدواره
        Console.WriteLine("================================")
        Dim S As Object
        For Each S In HT.Keys
            Console.WriteLine(S)
        Next
        Console.ReadKey()


      ' البحث داخل الهاش تيبل عن صنف بدلاله اسمه
        Console.WriteLine("Enter The Item Name ")
        Dim W = Console.ReadLine
        If HT.ContainsValue(W) Then
            Dim T = HT(W)
            Console.WriteLine(T)
            Console.ReadKey()
        Else
            Console.WriteLine("Item Not Found")
            Console.ReadKey()
        End If

        'البحث داخل الهاش تيبل عن صنف بدلاله مفتاحه
        Console.WriteLine("Enter The Item Key ")
        Dim Q = Console.ReadLine
        If HT.ContainsKey(Q) Then
            Console.WriteLine(HT(Q))
            Console.ReadKey()
        Else
            Console.WriteLine("Item Not Found")
            Console.ReadKey()
        End If
    End Sub
End Module
```

الشرح

اولا : قمنا باستيراد دوال الـ Collections لنستطيع استعمال الهاش تيبل
عن طريق الكود

```
Imports System.Collections
```

ثم قمنا بحجز متغير يحمل نسخه من خواص الهاش تيبل

```
Dim HT As New Hashtable
```

قولنا مسبقا ان الهاش تيبل يمكن ان يسجل بها المفتاح والبيان الخاص بالمفتاح باى نوع من الداتا لانه من النوع اوبجكت الذى يحمل اى قيم من البيانات ... لذلك قمت فى المثال السابق باضافه نوعين من انوع البيانات وهو نصى وحرفى لكى تفهم كيف تقوم بتسجيل كل نوع على حدى لانه تختلف طريقه التسجيل النصى عن الرقمى

فقمت باضافه بيان نصى ومفتاح رقمى بطريقتين عن طريق الكود

وهذه هى الطريقه الأولى باستخدام الميثود او الخاصيه Add
فهنا قمت بوضع المفتاح رقم 1 والبيان Mouse

```
HT.Add(1, "Mouse")
```

وهذه هى الطريقه الثانيه المفتاح رقمى والبيان نصى
فهنا قمت بوضع المفتاح رقم 2 والبيان sound , Monitor
لاحظ معى هنا انى وضعت اتنين بيان برقم مفتاح مشابه ... فهل سيأخذ المفتاح البيان الأول ام سيأخذ البيان الثانى ام سيأخذ الاثنين ؟ بالطبع سيأخذ البيان الثانى ويقوم بتلاشى الأول .. وهذا مايسمى بطريقه التعديل على بيان خاص بمفتاح معين ... اى ان اردنا ان نعدل على البيان ونضع بيان اخر نقوم بعمله عن طريق هذا الكود .

```
HT(2) = "Monitor"
HT(2) = "Sound"
```

وان اردنا ان نقوم بتسجيل مفتاح نصى وبيان نصى فيتم عن طريق الكود التالى

```
HT.Add("M", "Keyboard")
```

باختصار شديد ان الفرق بين اضافه مفتاح او بيان نصى و رقمى
ان النصى لابد ان يكتب بين دبل كوتيشن اى هذه العلامه ""
اما البيان الرقمى فهو يكتب حر بدون قيوم او علامات

وهنا طريقه عرض البيان على شاشه الكونسول بدلاله مفتاحه سواء كان نصى او رقمى
فكما قولنا ان الفارق الوحيد ان النصى يضع بين علامتي الدبل كوتيشن ولكن الرقمى يكتب حر بدون اى قيود او علامات وهذه قاعده عامه عند التعامل مع القيم النصيه فى اى مكان.

```
Console.WriteLine(HT(1))
Console.WriteLine(HT("M"))
```

وان اردنا ان نقوم بحذف بيان معين بدلاله مفتاحه فيكون عن طريق الكود

```
HT.Remove(1)
```

اى قم بمسح البيان الخاص بالمفتاح رقم 1 ... وتنطبق نفس الطريقه على المفتاح النصى ولكن لابد ان نراعى وضعه بين علامتى الدبل كوتيشن.

وان اردنا ان نقوم بعرض عدد المحتويات او البيانات المسجله بداخل الهاش تيبل فيكون ذلك عن طريق الكود التالى

```
Console.WriteLine("The item Count {0}", HT.Count)
```

اى قم بعرض عدد المحتويات التى بداخل الهاش تيبل مع افاده المستخدم ان هذا هو عدد الاصناف

وهنا لابد ان نقف وقفه صغيره لنعلق على الرقم المكتوب بجوار الكلمه The Item Count وهو الرقم صفر الذى بين البراكتس او القوسين .. هذا الرقم يفيد بان يعرض مابعد الكومه الأولى ... فان اردنا كتابه بيان اخر لابد من وضع كومه اخرى فما بعد الكومه الثانيه يساوى 1 فمن الممكن تغير الرقم لـ 1 ولكن هكذا ستكون الجمله بجوار البيان مابعد الكومه الثانيه.

وان اردنا عرض كل محتويات الهاش تيبل البيانات بدواره فتكون الدواره بسيطه جدا مثل مافعلنا بالامثله السابقه

```
Dim K As Object
For Each K In HT.Values
Console.WriteLine(K)
Next
```

طبعا احنا اتعودنا على اسلوب الدوارات ومش محتاج اشرحها تانى.

وان اردنا ان نعرض كل محتويات الهاش تيبل المفاتيح بدواره فتكون هى نفس فكره الدواره السابقه ولكن نجعله يدور حول المفاتيح ولا يدور حول القيم هكذا

```
Dim S As Object
For Each S In HT.Keys
Console.WriteLine(S)
Next
```


وان اردنا ان نقوم بالبحث داخل الهاش تيبل عن صنف معين بدلاله اسمه فيكون ذلك عن طريق الكود التالى :-

```
Console.WriteLine("Enter The Item Name ")
Dim W = Console.ReadLine
If HT.ContainsValue(W) Then
Dim T = HT(W)
Console.WriteLine(T)
Console.ReadKey()
Else
Console.WriteLine("Item Not Found")
Console.ReadKey()
End If
```


اولا: طلبت من الكونسول ان يعرض رساله للمستخدم يفيده بان يدخل اسم الصنف المراد البحث عنه

ثم قمت بحجز متغير ليحمل قيمه ما ادخل المستخدم

ثم قمت بعمل جمله شرطيه تقول .. ان كانت المصفوفه تحوى مابداخل المتغير(الذى يحمل قيمه ما ادخله المستخدم) اذن اعرض الناتج داخل متغير اخر
ثم اعرض مافى هذا المتغير على شاشه الكونسول .
وان لم تجد ما ادخله المستخدم Else
اعرض رساله للمستخدم على شاشه الكونسول تفيد بأنه لا يوجد صنف بهذا الأسم

فماذا ان اردنا ان نقوم بالبحث بدلاله رقم المفتاح ؟ ان فهمت فكره الكود الى فات هتعمل دى لوحدك .طيب تعالى نعملها سوى بس قوم بتطبيقها لوحدك الأول وان تعثرت فيها راجعها معايا.

```
Console.WriteLine("Enter The Item Key ")
Dim Q = Console.ReadLine
If HT.ContainsKey(Q) Then
Console.WriteLine(HT(Q))
Console.ReadKey()
Else
Console.WriteLine("Item Not Found")
Console.ReadKey()
End If
```


نفس فكره الكود السابق تماما
اولا: طلبت من الكونسول ان يعرض رساله للمستخدم يفيده بان يدخل رقم المفتاح المراد بدلالته
ثم قمت بحجز متغير ليحمل قيمه ما ادخل المستخدم
ثم قمت بعمل جمله شرطيه تقول .. ان كانت المصفوفه تحوى مابداخل المتغير(الذى يحمل قيمه ما ادخله المستخدم) اذن اعرض الناتج على شاشه الكونسول .
وان لم تجد ما ادخله المستخدم Else
اعرض رساله للمستخدم على شاشه الكونسول تفيد بأنه لا يوجد صنف بهذا المفتاح.

انتهت الوحده الرابعه :

الوحده الخامسه :-

SORTED LIST

السورتيد ليست .. هى ايضا نوع من انواع الكوليكشن وهى تشبه الهاش تيبل الى حد ما فهى تستخدم فى تسجيل صنف معين وتسجيل مفتاح مخصص لهذا الصنف مثل الهاش تيبل ولكن تختلف عنه بأنها تقوم بترتيب البيانات او الاصناف التى بداخلها تلقائيا فان قمنا مثلا بتسجيل صنف بمفتاح رقم 4 ثم قمنا بتسجيل صنف اخر بمفتاح برقم وليكن 3 فاننا عندما نأمرها بعرض مابداخلها نجدها تلقائيا تقوم بعرض اولا البيان الخاص بالمفتاح رقم 3 ثم البيان الخاص بالمفتاح رقم 4 .. وتتميز السورتيد ليست عن جميع انواع المصفوفات فى انها تبدأ بالاندكس رقم 1 .. وليس صفر كما هو الحال فى اغلب المصفوفات .

بعض الخصائص او الميثود الخاصه بالسورتيد ليست

الخاصيه Add

هذه الخاصيه تمكننا من اضافه بيان داخل السورتد ليست

الخاصيه Remove()

تمكننا هذه الخاصيه من مسح بيان معين داخل اجزاء السورتد ليست ويكون المسح بدلاله المفتاح حيث نقوم بوضع رقم المفتاح بين القوسين ان كان رقمى .. وان كان نصيا نقوم بوضعه بين القوسين وبين علامتي تنصيص فيكون هكذا Remove("MyKey")

خاصيه Values

لعرض قيمه معينه داخل اجزاء السورتد ليست

خاصيه Keys

لعرض مفتاح خاص بقيمه معينه داخل اجزاء السورتد ليست

خاصيه ContainsValue

تمكننا فى البحث داخل السورتد ليست عن قيمه بيان معين بداخل اجزائها

خاصيه ContainsKey

تمكننا فى البحث داخل السورتد ليست عن مفتاح خاص بقيمه معينه بداخل اجزائها

خاصيه Count

تمكننا بعرض عدد البيانات التى بداخل المصفوفه هاش تيبل

والأن سوف نتعرض الى مثال يوضح عمل الخواص او الميثود الخاصه بالسورتد ليست

تابع معى ...

```
Imports System.Collections
Module module1

    Sub main()

        Dim S As New SortedList
        S.Add(4, "Salma")
        S.Add(2, "Malek")
        S.Add(3, "Shafee")
        S.Add(1, "Nour")


        Dim X = S.Count
        Dim I As Integer
        For I = 1 To X
        Console.WriteLine("Number {0} = {1}", I, S(I))
        Next
        Console.ReadKey()

    End Sub
End Module
```


الشرح

اولا : قمنا باستيراد دوال الـ Collections لنستطيع استعمال الهاش تيبل عن طريق الكود

```
Imports System.Collections
```

ثم قمنا بحجز متغير من نوع المصفوفه سورتد ليست

```
Dim S As New SortedList
```

ثم قمنا باضافه بعض البيانات بارقام مفاتيح غير مرتبه فجعلنا البيان الاول مصاحب لمفتاح رقم 4 والثانى 2 والثالث 3 والرابع 1

لنرى كيف ستقوم المصفوفه ترتيبهم

```
S.Add(4, "Salma")
S.Add(2, "Malek")
S.Add(3, "Shafee")
S.Add(1, "Nour")
```

ثم قمنا بعمل دواره بسيطه تقوم بعرض محتويات المصفوفه على الترتيب

```
Dim X = S.Count
Dim I As Integer
For I = 1 To X
Console.WriteLine("Number {0} = {1}", I, S(I))
Next
```

اولا: قمنا بحجز متغير يحمل عدد البيانات التى بداخل السورتد ليست ولم نضع 1 - لاننا قولنا مسبقا ان هذه المصفوفه لاتحتوى على الاندكس رقم 0

ثم قمنا بحجز متغير اخر عددى ولنعرف فائده المتغيرين الأن
ثم بدأت الدواره ... اجعل الدوران يبدأ من رقم المتغير العددى (1) وينتهى بنهايه المتغير الذى يحمل عدد البيانات التى بداخل السورتد ليست
فى كل لفه اجعل شاشه الكونسول تكتب رقم اللفه ثم تكتب اسم البيان الخاص بهذه اللفه
ولاحظ معى هنا فى كود العرض

```
Console.WriteLine("Number {0} = {1}", I, S(I))
```

وضعنا رقم 0 و 1 بين قوسين .. فما سوف يعرض مكان الصفر هو مابعد الكومه الاولى وما سوف يعرض مكان الـ 1 او بعد العلامه = هو مابعد الكومه الثانيه `s(I)`

فلك ان تتخيل الناتج لكى يتضح لك العمليه اكثر
لاحظ الصوره



فان قمت بمراجعه الكود فستجد ان البيان الأول المسجل بالمصفوفه هو البيان Salma
ولكن لأنه مصاحب للمفتاح الأكبر قيمه فكان اخر بيان معروض فى شاشه الكونسول

كيفيه البحث وعرض عدد الأصناف وباقى المميزات على نمط الهاش تيبل

انتهت الوحده الخامسه :

الوحده السادسه :-

STRING DICTIONARY

الاسترينج ديكشنرى هى ايضا من انواع الكوليكشن ولكنها تختص بجزء بداخل الكوليكشن يسمى Specialized.

لذلك لابد عند استعمالها استيرادها عن طريق الكود

```
Imports System.Collections.Specialized
```

وهى تشبه عمل السورتيد ليست فهى تقوم بترتيب بياناتها تلقائيا ولكنها تختلف عنها فى انها تستخدم فى اضافه بيانات ومفتاح خاص بكل بيان ولكن لاتقبل الا القيم النصيه سواء كان فى المفتاح ا وفى البيان فهى لاتقبل القيم الرقميه او التاريخ اوالمزدوجه ... الخ. لأنها من النوع string ولذلك يطلق عليها

string Dictionary

وتقوم بترتيب بياناتها بدلاله المفاتيح فان قمنا بادخال بيان بالمفتاح D

ثم قمنا بادخال بيان اخر بالمفتاح A

فانها عند العرض تقوم اولا بعرض البيان الخاص بالمفتاح A ثم المفتاح D

بعض الخصائص او الميثود الخاصه بالأسترينج ديكشنرى

الخاصيه Add

هذه الخاصيه تمكننا من اضافه بيان داخل الاسترينج ديكشنرى

الخاصيه Remove()

تمكننا هذه الخاصيه من مسح بيان معين داخل اجزاء الاسترينج ديكشنرى ويكون المسح بدلاله المفتاح حيث نقوم بوضعه بين القوسين وبين علامتي تنصيص فيكون هكذا

```
Remove("MyKey")
```

خاصيه Values

لعرض قيمه معينه داخل اجزاء الاسترينج ديكشنرى

خاصيه Keys

لعرض مفتاح خاص بقيمه معينه داخل اجزاء الاسترينج ديكشنرى

والأن سوف نتعرض الى مثال يوضح عمل الخواص او الميثود الخاصه بالاسترينج ديكشنرى

تابع معى ...

```vb
Imports System.Collections.Specialized
Module module1

    Sub Main()

        Dim SD As New StringDictionary
        SD.Add("E", "Eat")
        SD.Add("D", "Drink")
        SD.Add("A", "Apple")
       'SD.Remove("A")

        Console.WriteLine("==============Values=================")
        Dim S As String
        For Each S In SD.Values
            Console.WriteLine(S)
        Next
        Console.ReadKey()


        Console.WriteLine("============Keys===================")
        Dim K As String
        For Each K In SD.Keys
            Console.WriteLine(K)
        Next
        Console.ReadKey()

    End Sub

End Module
```

الشرح

اولا : قمنا باستيراد دوال الـ Collections لنستطيع استعمال الاسترينج ديكشنرى عن طريق الكود

```vb
Imports System.Collections.Specialized
```

ثم قمنا بحجز متغير من نوع المصفوفه سترينج ديكشنرى

```vb
Dim SD As New StringDictionary
```

ثم قمنا باضافه بيانات داخل الاسترينج ديكشنرى باسماء مفاتيح وطريقه غير مرتبه لنرى كيف ستقوم المصفوفه سترينج ديكشنرى بترتيب هذه البيانات بدلاله مفاتيحها

```vb
SD.Add("E", "Eat")
SD.Add("D", "Drink")
SD.Add("A", "Apple")
```

لاحظ معى فى الصوره التاليه كيف قامت بترتيب البيانات :-



ثم ان اردنا ان نقوم بحذف بيان معين بدلاله مفتاحه فيكون كالأتى :-

```
SD.Remove("A")
```

اى قم بحذف البيان الذى يختص بالمفتاح A

ثم قمنا بعمل دواره تقوم بعرض كل محتويات الاسترينج ديكشنرى القيم

```
Dim S As String
For Each S In SD.Values
Console.WriteLine(S)
Next
```

تفيد هذه الدواره باللف حول المتغير النصى s
طالما كان هذا المتغير يحوى قيمه من القيم الموجوده داخل الاسترينج ديكشنرى
وفى كل لفه تعرض محتوى من محتويات الاسترينج ديكشنرى حتى تصل لنهايه المحتويات فينتهى عمل الدواره

فماذا ان اردنا ان نقوم بعمل دواره اخرى تقوم بعرض كل محتويات الاسترينج ديكشنرى المفاتيح ؟ بالتأكيد ان فهمت عمل الدواره السابقه ستقوم بعمل هذه الدواره بمفردك
اذن قم بالتجربه هل تستطيع عملها ام لا . فان تعثرت فتابع معى

```
Dim K As String
For Each K In SD.Keys
Console.WriteLine(K)
Next
```

تفيد هذه الدواره باللف حول المتغير النصى k
طالما كان هذا المتغير يحوى مفتاح من المفاتيح الموجوده داخل الاسترينج ديكشنرى
وفى كل لفه تعرض مفتاح من المفاتيح المسجله داخل الاسترينج ديكشنرى حتى تصل لنهايه المفاتيح فينتهى عمل الدواره

انتهت الوحده السادسه :

الوحده السابعه :-

LIST DICTIONARY

الليست ديكشنرى هى ايضا من انواع الكوليكشن ولكنها تختص بجزء بداخل الكوليكشن يسمى Specialized.

لذلك لابد عند استعمالها استيرادها عن طريق الكود

```
Imports System.Collections.Specialized
```

وهى تماما مثل عمل الهاش تيبل لاتختلف عنه فى اى شىء.. الا انها اسرع منه فى التعامل ولكنها لا تحمل الا عشره بيانات فقط فان زاد عليها اكثر من عشره بيانات لاتعمل.
ويفضل العمل بها عن العمل بالهاش تيبل ولكن ان تأكدنا بأن الأصناف الى سوف تسجل بها سوف لن تزيد عن عشره اصناف. واجزائها تحمل جميع انواع الداتا سواء كانت المفاتيح او القيم الخاصه بكل مفتاح فهى من النوع اوبجكت كما هو الحال تماما فى الهاش تيبل ... لمعرفه استخدام الهاش تيبل راجع الوحده الرابعه .

بعض الخصائص او الميثود الخاصه بالليست ديكشنرى

الخاصيه Add
هذه الخاصيه تمكننا من اضافه بيان داخل الليست ديكشرى

الخاصيه ()Remove
تمكننا هذه الخاصيه من مسح بيان معين داخل اجزاء المصفوفه ليست ديكشنرى ويكون المسح بدلاله المفتاح حيث نقوم بوضع رقم المفتاح بين القوسين ان كان رقمى .. وان كان نصيا نقوم بوضعه بين القوسين وبين علامتي تنصيص فيكون هكذا Remove("MyKey")

خاصيه Values
لعرض قيمه معينه داخل اجزاء الليست ديكشنرى

خاصيه Keys
لعرض مفتاح خاص بقيمه معينه داخل اجزاء الليست ديكشنرى

خاصيه ContainsValue
تمكننا فى البحث داخل الليست ديكشنرى عن قيمه بيان معين بداخل اجزائها

خاصيه ContainsKey
تمكننا فى البحث داخل الليست ديكشنرى عن مفتاح خاص بقيمه معينه بداخل اجزائها

خاصيه Count
تمكننا بعرض عدد البيانات التى بداخل الليست ديكشنرى

والأن سوف نتعرض الى مثال يوضح عمل الخواص او الميثود الخاصه بالليست ديكشنرى
تابع معى ...

```vb
Imports System.Collections.Specialized
Module module1
    Sub main()

        Dim LD As New ListDictionary
        LD.Add(1, "SPeaker")
        LD.Add(2, "Mouse")
        LD.Add(3, "Monitor")
        LD.Add(4, "Keyboard")



        Console.WriteLine("===========Keys=============")
        Dim K As Object
        For Each K In LD.Keys
            Console.WriteLine(K)
        Next


        Console.WriteLine("===========Values===========")
        Dim V As Object
        For Each V In LD.Values
            Console.WriteLine(V)
        Next



        Console.WriteLine("===========Keys & Values===========")
        Console.WriteLine("----------------")
        Console.WriteLine("Keys   | Values")
        Console.WriteLine("----------------")

        Dim KV As DictionaryEntry
        For Each KV In LD
            Console.WriteLine(" {0}     | {1} ", KV.Key, KV.Value)
        Next
        Console.WriteLine("----------------")
        Console.ReadKey()


    End Sub
End Module
```


الشرح

اولا : قمنا باستيراد دوال الـ Collections لنستطيع استعمال الاسترينج ديكشنرى عن طريق الكود

```vb
Imports System.Collections.Specialized
```

ثم قمنا بحجز متغير من نوع المصفوفه ليست ديكشنرى

```vb
Dim LD As New ListDictionary
```

ثم قمنا باضاف بعض البيانات بداخلها مفاتيح من النوع الرقمى وبيانات من النوع النصى

```
LD.Add(1, "SPeaker")
LD.Add(2, "Mouse")
LD.Add(3, "Monitor")
LD.Add(4, "Keyboard")
```

ثم قمنا بعمل دواره تقوم بعرض مابداخل الليست ديكشنرى من مفاتيح

```
Dim K As Object
For Each K In LD.Keys
Console.WriteLine(K)
Next
```

قمنا بحجز متغير من النوع اوبجكت وذلك ضمانا لعدم حدوث خطأ وافتراضا اننا لانعلم ماهى نوع الداتا للمفاتيح التى بداخل الليست ديكشنرى .. فقمنا بحجز المتغير اوبجكت ليقبل جميع البيانات .
ثم بدأت الدواه .. تدور طالما كان هذا المتغير يحوى مفتاح من مفاتيح الليست ديكشنرى وفى كل لفه تقوم بعرض مفتاح حتى تقوم بعرض كل المفاتيح فينتهى عمل الدواره

فما ذا ان اردنا ان نقوم بعمل دواره تعرض كل محتويات الليست ديكشنرى من قيم ؟ بالطبع ان فهمت الدواره السابقه فستقوم بعمل هذه الدواره بمفردك ... جرب وقم بعملها بمفردك دون النظر الى اكوادها او الاطلاع عليها ... فان تعثرت فى عملها تابع معى

```
Dim V As Object
For Each V In LD.Values
Console.WriteLine(V)
Next
```

قمنا بحجز متغير من النوع اوبجكت ايضا وذلك ضمانا لعدم حدوث خطأ وافتراضا اننا لانعلم ماهى نوع الداتا للقيم التى بداخل الليست ديكشنرى .. فقمنا بحجز المتغير اوبجكت ليقبل جميع البيانات سواء كانت نصيه او رقميه او مزدوجه ... الخ.
ثم بدأت الدواه .. تدور طالما كان هذا المتغير يحوى بيان من بيانات الليست ديكشنرى وفى كل لفه تقوم بعرض بيان حتى تقوم بعرض كل البيانات فينتهى عمل الدواره.

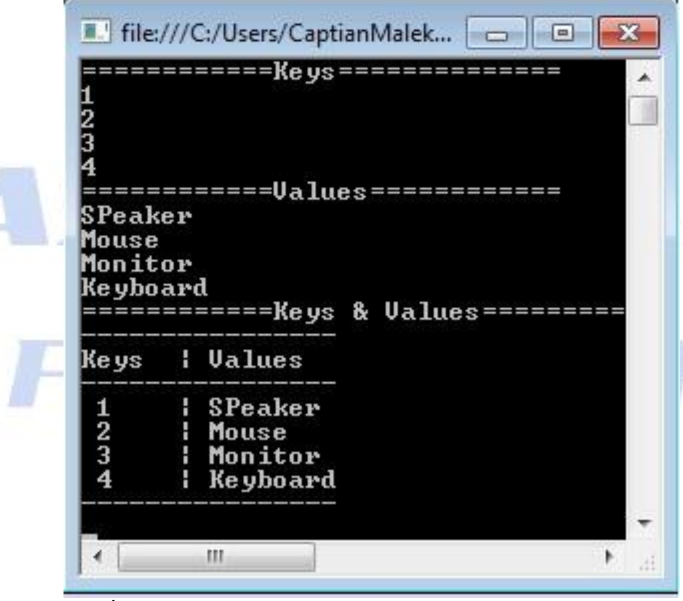
فما ذا ان اردنا ان نقوم بعمل دواره تعرض كل محتويات الليست ديكشنرى من قيم ومفاتيح كل مفتاح بجواره القيمه الخاصه به ؟ الطريقه ابسط مما تتخيل فلا تدع الكلام يهيء فى عقلك ان الفكره كبيره او مستحيله او حتى انها صعبه .. فقط ركز معى وسوف تعلم كيف نقوم بعمل هذه الدواره .

```
Dim KV As DictionaryEntry
For Each KV In LD
Console.WriteLine(" {0}     | {1} ", KV.Key, KV.Value)
Next
Console.WriteLine("---------------")
```

قمنا بحجز متغير من النوع DictionaryEntry
هذا النوع هو نوع يحوى مابداخل المصفوفات التى من النوع الديكشنرى وهو يحوى مابداخلها من مفاتيح واصناف ... فيمكن استعماله مع اى نوع من انواع الديكشنرى فهو افضل من ان

نقوم بحجز متغير اوبجكت لانه يكون اسرع منه بكثير... لأن النوع اوبجكت فى الميمورى يقوم بالعمل عن طريق البحث عن كل بيان ثم ان كان هذا البيان من النوع النصى يقوم بحجز متغير من النوع النصى فى الميمورى وان كان رقمى يقوم بحجز متغير رقمى فى الميمورى ...
وهكذا ... فهو يأخذ وقت ولن نشعر بهالا فى البيانات الضخمه لذلك لابد ان نراعى هذه النقاط الصغيره المهمه .
ثم بدأت الدواه .. تدور طالما كان هذا المتغير يحوى بيان كامل من بيانات الليست ديكشنرى ولم نقم بالشرط عليه بقيمه صنف او مفتاح .
وفى كل لفه تقوم بعرض بيان كامل مفتاح وصنف حتى تقوم بعرض كل البيانات فينتهى عمل الدواره.

وهذه هى النتيجه فى الصوره التاليه :-



لاحظ ان النتيجه هى عرض القيم المفاتيح فقط ... وهو ناتج الدواره الأولى
وعرض القيم الاصناف فقط... وهو ناتج الدواره الثانيه
وعرض القيم الاصناف والمفاتيح .. وهو ناتج الدواره الثالثه

انتهت الوحده السابعه :

الوحده الثامنه :-

HYBRID DICTIONARY

الهايبرد ديكشنرى هى ايضا من انواع الكوليكشن ولكنها تختص بجزء بداخل الكوليكشن يسمى Specialized.
لذلك لابد عند استعمالها استيرادها عن طريق الكود

```
Imports System.Collections.Specialized
```

...ولقد تطرقنا للقول مسبقا انه يفضل العمل باالليست ديكشنرى ان تأكدنا بأن الأصناف التى ستضاف به سوف لن تزيد عن عشر أصناف وان زادت عن ذلك فلابد من استعمال الهاش تيبل لأنه يحمل عدد اصناف اكثر من الليست ديكشنرى .. ولكن الهايبرد ديكشنرى تقوم بالتحويل بين الليست ديكشنرى والهاش تيبل فهى عندما تكون اصنافها اقل من عشره اصناف فهى تقوم باستخدام الليست ديكشنرى وان زادت الأصناف عن عشره تقوم تلقائيا بتحويل نفسها على نظام الهاش تيبل . فهى افضل فى الأستخدام فى حاله الزعزعه فى عدد الأصناف .. ولذلك ينصح بأستخدامها . والتعامل معها هو نفس التعامل مع الليست ديكشنرى والهاش تيبل تماما لا يوجد أى اختلاف.

بعض الخصائص او الميثود الخاصه بالهايبرد ديكشنرى

الخاصيه Add
هذه الخاصيه تمكننا من اضافه بيان داخل الهايبرد ديكشرى

الخاصيه Remove()
تمكننا هذه الخاصيه من مسح بيان معين داخل اجزاء المصفوفه هايبرد ديكشنرى ويكون المسح بدلاله المفتاح حيث نقوم بوضع رقم المفتاح بين القوسين ان كان رقمى .. وان كان نصيا نقوم بوضعه بين القوسين وبين علامتي تنصيص فيكون هكذا Remove("MyKey")

خاصيه Values
لعرض قيمه معينه داخل اجزاء الهايبرد ديكشنرى

خاصيه Keys
لعرض مفتاح خاص بقيمه معينه داخل اجزاء الهايبرد ديكشنرى

خاصيه ContainsValue
تمكننا فى البحث داخل الهايبرد ديكشنرى عن قيمه بيان معين بداخل اجزائها

خاصيه ContainsKey
تمكننا فى البحث داخل الهايبرد ديكشنرى عن مفتاح خاص بقيمه معينه بداخل اجزائها

خاصيه Count
تمكننا بعرض عدد البيانات التى بداخل الهايبرد ديكشنرى

والأن سوف نتعرض الى مثال يوضح عمل الخواص او الميثود الخاصه الهايبرد ديكشنرى تابع معى ...

```
Imports System.Collections.Specialized
Module module1

    Sub Main()
        Dim HD As New HybridDictionary
        HD.Add(1, "Mouse")
        HD.Add(2, "Keyboard")
        HD.Add(3, "Sound")


        Console.WriteLine("===========Keys==========")
        Dim K As DictionaryEntry
        For Each K In HD
            Console.WriteLine(K.Key)
        Next

        Console.WriteLine("===========Values==========")
        Dim V As DictionaryEntry
        For Each V In HD
            Console.WriteLine(V.Value)
        Next


        Console.WriteLine("===========Keys & Values==========")
        Dim KV As DictionaryEntry
        For Each KV In HD
            Console.WriteLine("{0} {1}", KV.Key, KV.Value)
        Next
        Console.ReadKey()

    End Sub
End Module
```


الشرح

اولا : قمنا باستيراد دوال الـ Collections لنستطيع استعمال الهايبرد ديكشنرى عن طريق الكود

```
Imports System.Collections.Specialized
```

ثم قمنا بحجز متغير من نوع المصفوفه ليست ديكشنرى

```
Dim HD As New HybridDictionary
```


ثم قمنا باضافه ثلاث بيانات فقط بداخل الهايبرد ديكشنرى ففى هذه الحاله تلقائيا سوف تقوم بالعمل مع الليست ديكشنرى اما ان قومنا بتزويد البيانات حتى تزايدت عن عشره بيانات كما قولنا سوف تتعامل على انها هاش تيبل

```
HD.Add(1, "Mouse")
```

```
HD.Add(2, "Keyboard")
HD.Add(3, "Sound")
```

ثم قمنا بعمل دواره تقوم بعرض مابداخل الليست ديكشنرى من مفاتيح

```
Dim K As DictionaryEntry
For Each K In HD
Console.WriteLine(K.Key)
Next
```

قمنا بحجز متغير من نوع البيانات التى بداخل الهايبرد ديكشنرى
ثم بدأت الدواره .. تدور طالما كان هذا المتغير يحوى بيان من بيانات الهايبرد ديكشنرى
وفى كل لفه تقوم بعرض مفتاح حتى تقوم بعرض كل المفاتيح فينتهى عمل الدواره

فما ذا ان اردنا ان نقوم بعمل دواره تعرض كل محتويات الهايبرد ديكشنرى من قيم ؟ بالطبع ان فهمت الدواره السابقه فستقوم بعمل هذه الدواره بمفردك ... جرب وقم بعملها بمفردك دون النظر الى اكوادها او الاطلاع عليها ... فان تعثرت فى عملها تابع معى

```
Dim V As DictionaryEntry
For Each V In HD
Console.WriteLine(V.Value)
Next
```

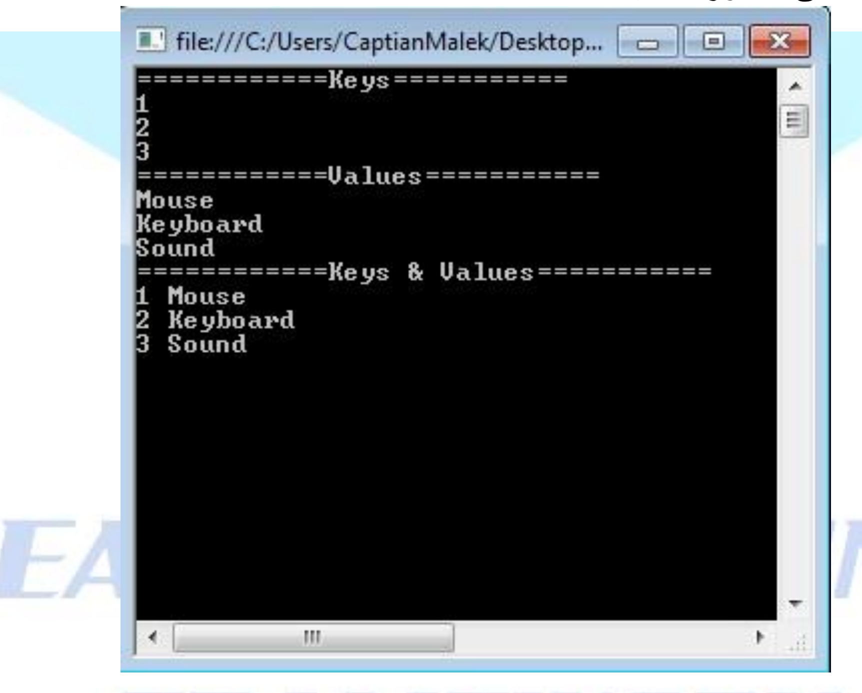
قمنا بحجز متغير من نوع البيانات التى بداخل الهايبرد ديكشنرى
ثم بدأت الدواره .. تدور طالما كان هذا المتغير يحوى بيان من بيانات الهايبرد ديكشنرى
وفى كل لفه تقوم بعرض صنف حتى تقوم بعرض كل الأصناف التى بداخلها فينتهى عمل الدواره .

فما ذا ان اردنا ان نقوم بعمل دواره تعرض كل محتويات الهايبرد ديكشنرى من قيم ومفاتيح كل مفتاح بجواره القيمه الخاصه به ؟ الطريقه هى نفس الطريقه التى قمنا بعملها فى الليست ديكشنرى تماما تابع معى.

```
Dim KV As DictionaryEntry
For Each KV In HD
Console.WriteLine("{0} {1}", KV.Key, KV.Value)
Next
```

قمنا بحجز متغير من النوع DictionaryEntry
هذا النوع هو نوع يحوى مابداخل المصفوفات التى من النوع الديكشنرى و هو يحوى مابداخلها من مفاتيح واصناف .
ثم بدأت الدواه .. تدور طالما كان هذا المتغير يحوى بيان كامل من بيانات الهايبرد ديكشنرى ولم نقم بالشرط عليه بقيمه صنف او مفتاح .
وفى كل لفه تقوم بعرض بيان كامل مفتاح وصنف حتى تقوم بعرض كل البيانات فينتهى عمل الدواره.

وهذه هى النتيجه فى الصوره التاليه :-



انتهت الوحده الثامنه :



الوحده التاسعه :-

NAME VALUE COLLECTION

النيم فاليو كوليكشن ...هى ايضا من انواع الكوليكشن ولكنها تختص بجزء بداخل الكوليكشن يسمى Specialized.
لذلك لابد عند استعمالها استيرادها عن طريق الكود

```
Imports System.Collections.Specialized
```

ولها استعمال مختلف عن ما قبلها من انواع المصفوفات فى الكوليكشن .. ويعد لها اهميه كبيره فى السوفت وير الضحم... فهى تقوم بعمل تصنيف لبياناتها على شكل فئات .. فلو تخيلنا مثلا استخدامها فى سوفت وير لمعرض سيارات .. فان كان فى المعرض نوع معين من العربيات وله اكثر من اصدار فيمكننا استخدامها فى تصنيف كل نوع وكل اصدار من هذا النوع على شكل فئات . وهى لا تقبل الا البيانات النصيه فقط فى المفاتيح والأصناف .
فهى تشبه الـstring Dictionary فى هذه الخاصيه

بعض الخصائص او الميثود الخاصه بالنيم فاليو كوليكشن

الخاصيه Add
هذه الخاصيه تمكننا من اضافه بيان داخل النيم فاليو كوليكشن

الخاصيه Remove()
تمكننا هذه الخاصيه من مسح بيان معين داخل اجزاء المصفوفه نيم فاليو كوليكشن ويكون المسح بدلاله المفتاح نقوم بوضعه بين القوسين وبين علامتي تنصيص فيكون هكذا

```
Remove("MyKey")
```

الخاصيه GetValues
لعرض القيم التى بداخل النيم فاليو كوليكشن من اصناف

الخاصيه AllKeys
لعرض كل القيم التى بداخل النيم فاليو كوليكشن من مفاتيح

والأن سوف نتعرض الى مثال يوضح عمل الخواص او الميثود الخاصه بالنيم فاليو كوليكشن
تابع معى ...

```
Imports System.Collections.Specialized
Module module1
    Sub main()

        Dim NVC As New NameValueCollection
        NVC.Add("Fiat", "Fiat 128")
        NVC.Add("Fiat", "Fiat 132")
        NVC.Add("Fiat", "Tempra")
        NVC.Add("Seat", "Ibiza")
        NVC.Add("Seat", "Lion")
        NVC.Add("Seat", "Kortoba")
        NVC.Add("Toyota", "Corola")
        NVC.Add("Toyota", "Yaris")
        NVC.Add("Hyundai", "Accent")
        NVC.Add("Hyundai", "Elantra ")
        NVC.Remove("Fiat")



        Console.Write("Enter The Kind Name Of Car: ")
        Dim S = Console.ReadLine
        Console.WriteLine(NVC(S))
        Console.ReadKey()


        Console.WriteLine("==========BY Loop======")
        Dim Kind As String
        For Each Kind In NVC.GetValues("Fiat")
            Console.WriteLine(Kind)
        Next Kind
        Console.ReadKey()


        Console.WriteLine("=======Kind of Car==========")
        Dim K As String
        For Each K In NVC.AllKeys
            Console.WriteLine(K)
        Next K
        Console.ReadKey()


    End Sub
End Module
```


الشرح

اولا : قمنا باستيراد دوال الـ Collections لنستطيع استعمال النيم فاليو كوليكشن عن طريق الكود

```
Imports System.Collections.Specialized
```

ثم قمنا بحجز متغير من نوع المصفوفه نيم فاليو كوليكشن

```
Dim NVC As New NameValueCollection
```

ثم قمنا باضافه بعض انواع السيارات على اعتبار اننا نقوم بعمل سوفت وير لمعرض سيارات فقمنا باضافه بعض انواع السيارات والموديلات الموجوده منها فى هذا المعرض

```
NVC.Add("Fiat", "Fiat 128")
NVC.Add("Fiat", "Fiat 132")
NVC.Add("Fiat", "Tempra")
NVC.Add("Seat", "Ibiza")
NVC.Add("Seat", "Lion")
NVC.Add("Seat", "Kortoba")
NVC.Add("Toyota", "Corola")
NVC.Add("Toyota", "Yaris")
NVC.Add("Hyundai", "Accent")
NVC.Add("Hyundai", "Elantra ")
```

لاحظ ان المفتاح مكرر فى اكثر من بيان ... وهذه خاصيه تنفرد بها النيم فاليو كوليكشن فانك لو راجعت كل المصفوفات السابقه ستجد انه لا يوجد اى مصفوفه تدعم تكرار المفتاح وان قمت بتكرار المفتاح تعتبره تعديل على ماقبله ..
فهنا قمنا باضافه انواع السيارات ولكل نوع اكثر من موديل او اصدارى

فماذا لو أراد صاحب المعرض ان يستعلم عن الموديلات الموجوده من نوع معين عنده ؟
تابع معى الكود

```
Console.Write("Enter The Kind Name Of Car: ")
Dim S = Console.ReadLine
Console.WriteLine(NVC(S))
```

قمنا بأمر الكونسول ان يعرض رساله لصاحب المعرض يأمره فيها بأدخال نوع السياره المراد الاستعلاام عن موديلاتها او اصداراتها.
ثم وضعنا قيمه ماكتبه صاحب المعرض فى متغير .. اذا اصبح هذا المتغير يحمل قيمه مايريده صاحب المعرض .
ثم امرنا الكونسول بكتابه كل الاصدارات المسجله لهذا النوع .. اذا كيف يتم عمل هذا داخل المصفوفه ؟ .. تقوم المصفوفه بالدوارن حول محورها وتأخذ القيم المشتركه فى مفتاح معين اى ان كتب صاحب المعرض فيات .. درات حول محورها لتأخذ كل ماهو يخص المفتاح المسمى فيات ثم تقوم بعرضه على شاشه الكونسول كما امرناها .
اعتقد ان العمليه اصبحت سهله الاستيعاب الأن !!

السؤال الأن .. هل يوجد فكره اخرى لعمل الكود السابق ؟ بدواره مثلا ؟ هل تستطيع فعل ذلك ؟ بالطبع لو تألق تركيزك معى فى الدروس السابقه ستعرف كيف يتم فعل ذلك بدواره ..
اذا اذا رأيت نفسك تستطيع عمل فكره الدواره قم بعملها .. ولكن عليك ان لا تتخيلها بشكل كبير اجووووو منك اخى القارىء ان تنظر الى اى شىء فى البرمجه انه ممكن وليس مستحيلا وليس صعبا.. اذا تعثرت عمل فكره الكود السابق بدواره قم باتباع الشرح التالى :-

```
Dim Kind As String
For Each Kind In NVC.GetValues("Fiat")
Console.WriteLine(Kind)
Next Kind
```

هذه فكره دواره بسيطه تقوم بالدوارن حول موديل فيات وتكشف مابه من اصدارات
قمنا بحجز متغير فى الميمورى من النوع النصى
وبدأت الدواره تدور على اساس وجود هذا المتغير فى الـقيم الخاصه بالموديل فيات فى كل لفه
ففى كل لفه تقوم باللف حول محور المصفوفه نيم فاليو كوليكشن حتى تستنبط منها كل العوامل المشتركه فى المفتاح او الموديل المسمى فيات والذى قمنا بتحديده لها .. ثم تقوم بعرضه على شاشه الكونسول.
فكره بسيطه ؟ ولكن هذا الدواره ثابته النتائج اى انها تدول حول محور الموديل المسمى فيات فقط .. فهل هذا منطقى ؟ هل صاحب المعرض سوف لن يستعلم الا عن موديل فيات ؟ اعتقد انك لو قومت بتقديم برنامج مثل هذا لصاحب معرض .. جميل ان قام بطردك فقط !!!
اذا كيف نجعل صاحب المعرض او مستخدم البرنامج يقوم بتحديد النوع ؟
بكل بساطه ... قبل ان ننشأ الدواره نقوم بحجز متغير هذا المتغير يكون من النوع النصى لأننا نعلم ان المصفوفه نيم فاليو كوليكشن لاتقبل الا النصوص فقط .
ثم نأمر الكونسول ان يأمر المستخدم بكتابه اسم الموديل المراد البحث عنه ونضع قيمه ما كتبه المستخدم فى هذا المتغير النصى ... ثم عند امر العرض نضع قيمه المتغير بدلا من كلمه فيات فى الدواره السابقه .. ان لم تصل الفكره اليك فدعنى اكتب لك الأكواد لتضح اليك اكثر .

```
Console.Write("Enter The Model Where You Like To Query : ")
Dim Ki As String = Console.ReadLine
Dim Kind As String
For Each Kind In NVC.GetValues(Ki)
Console.WriteLine(Kind)
Next Kind
```

ماذا فعلت هنا؟ كل ما اضفته على الكود السابق انى قمت بحجز متغير يحمل قيمه مايريده المستخدم ثم امرت الكونسول فى كل لفه ان يعرض القيم الخاصه بما ادخله المستخدم.

سؤال اخر .. ماذا لو اراد صاحب المعرض ان يعرض كل الأنواع الموجوده عنده من سيارات؟ اى فيات مثلا وتويوتا وهيونداى .. الخ فكيف يتم ذلك ؟ تابع معى
لاتفزع من شكل السؤال او الطلب فهو بكل بساطه يريد عرض الانواع الخاصه بالسيارات الموجوده فى المعرض .. اين تسجل هذه الأنواع ؟ فى المصفوفه مكان المفتاح جميل هل هو صعب ان تقوم بعرض كل المفاتيح الموجوده فى المصفوفه ؟ لقد قمنا بعرضها مسبقا فى كل المصفوفات السابقه وكانت الفكره بسيطه جدا ... ولكن هذه المصفوفه يوجد بها تكرار فى المفاتيح فهل عند العرض سوف تقوم بعرض النتائج مكرره ؟ بالطبع لا هى تقوم تلقائيا بعمل Distinct لمفاتيحها لكى لاتعرض بطريقه مكرره.
اذا دعنا نرى كيفيه عرض المفاتيح فى النيم فاليو كوليكشن .

```
Dim K As String
For Each K In NVC.AllKeys
Console.WriteLine(K)
Next K
```

قمنا بحجز متغير من النوع النصى
وقومنا بعمل دواره تقوم بعرض كل المفاتيح الموجوده فى المصفوفه نيم فاليو كوليكشن فى كل لفه حتى تنتهى الدواره من عرض كل مفاتيح المصفوفه فيتنهى عملها

اعلم عزيزى الطالب!!!

ا ن الدواره السابقه قائمه على العرض من خلال الميثود او الخاصيه التى تسمى AllKeys هذه الخاصيه او الميثود هو تلقائيا يقوم بفلتره المفاتيح قبل عرضها بحيث لا يقوم بعرضها مكرره وهو مانوهنا اليه مسبقا.

انظر الى نتائج الأكواد السابقه فى الصوره التاليه :-





انتهت الوحده التاسعه :

الوحده العاشره :-

GENERIC COLLECTION

الجنيرك كوليكشن .. هوليس بمصفوفه ولا يحمل خواص المصفوفات .. ولكن هو عباره عن كلاس داخل الكوليكشن يسمى جنيرك ... فما هى فائدته او ماهو عمله ؟
فائدته كبيره جدا وتظهر فى المشاريع الضخمه التى تحمل البيانات بالالاف ... حيث انه يجعلك تحدد نوع الداتا الداخله لأى مصفوفه حيث انك اذا حددت هذه الداتا فلايمكنك بيان من نواع اخر داخلها غير المحدد ... بطريقه اوضح تخيل معى ماذا كنا نفعل لحجز مصفوفه مثلا من النوع كيو ؟ كنا نقول `Dim QU As New Queue`
معنى هذا الكود انك قمت بحجز مخزن فى الميمورى نسخه من المصفوفه كيو يحمل خصائصها كامله .. ولكن يمكننى داخل هذه المصفوفه الأن ان اسجل بها اى نوع من انواع الداتا صحيح ؟
انك عندماقمت بحجز هذا المخزن فى الميمورى هو لم يعلم اى نوع من انواع الداتا سوف تدخل فى المصفوفه فهو يظل معلقا حتى تعطى له نوع الداتا ثم يقوم بحجزها فى الميمورى بنوعها لكننا نعلم ان هذا حمل على الميمورى وضياع للوقت ولكن لن نشعر به فى ظل البيانات الصغيره .
ولكن يمكنك الان تحديد نوع الداتا او البيانات التى يجب ان تدخل فى المصفوفه عن طريق استخدام الجنيرك كوليكشن . فتخيل ان قمنا بحجز مخزن بالجنيرك كوليكشن كيف ستكون الطريقه ؟ `Dim QU As New Queue(Of String )`
تخيلت ماذا فعل الجنيرك فى هذا الكود ؟ هو حدد ان يكون نوع المصفوفه من النوع النصى ولن يسمح بدخول النوع الرقمى او المزدوج او اى داتا من الانواع الأخرى .
والأن نستطيع ان نقوم بتعريف الجنيرك .. هو كلاس من المكتبه كوليكشن يسمى جنيرك ولذلك عند استخدامه لابد من استيراده عن طريق الكود

```
Imports System.Collections.Generic
```

وهو وسيله للتسهيل فى الميمورى عند حجز مصفوفه من نوع معين .

والأن هيا بنا نأخذ مثال يوضح ماقمنا بشرحه:-

```
Imports System.Collections.Generic
Module module1

    Sub main()

        Dim QU As New Queue(Of String)
        QU.Enqueue(5)
        QU.Enqueue(5)
        Console.WriteLine(QU.Dequeue + QU.Dequeue)
        Console.ReadKey()


        Dim QU2 As New Queue(Of Integer)
        QU2.Enqueue(10)
        QU2.Enqueue(20)
        Console.WriteLine(QU2.Dequeue + QU2.Dequeue)
        Console.ReadKey()

    End Sub

End Module
```


الشرح

اولا : قمنا باستيراد دوال الـ Collections لنستطيع استعمال الجنيرك كوليكشن عن طريق الكود

```
Imports System.Collections.Generic
```

ثم قمنا بحجز مصفوفه من النوع كيو ولكن على شرط انها تحجز فى الميمورى انها سوف تتعامل مع بياناتها على اساس انها من النوع النصى فقط .. لمعرفه كيفيه عمل الكيو بالتفصيل راجع الوحده الثانيه

```
Dim QU As New Queue(Of String)
```

لك ان تتخيل الفرق بين طريقه حجزها بالجنيرك وطريقه حجزها بالطريقه العاديه ولقد شرحنا ماهى ميكانيكيه العمل بالطريقتين فى اول الوحده . ولولا انك قمت باستيراد الكلاس الخاص بالجنيرك لم تستطيع حجز المصفوفه على هذا الشرط Of string

لا تشتت تفكيرك ودعنا نكمل
ثم قمنا باضافه بيانات من النوع الرقمى داخل المصفوفه ...

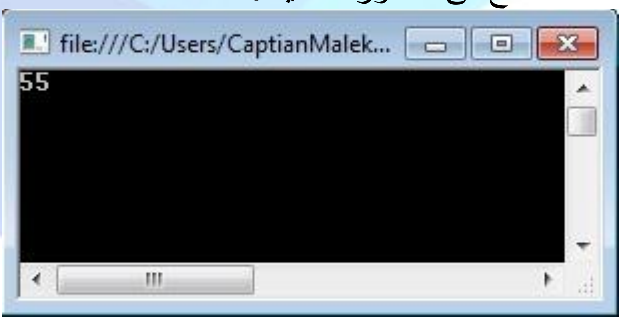
```
QU.Enqueue(5)
QU.Enqueue(5)
```


كيف هذا ؟ انت قولت لنا انك قومت بحجزها على اساس انها نصيه وليست رقميه ثم قمت باضافه بيانات من النوع الرقمى فكيف ستتعامل معها المصفوفه الأن ؟ هل سيتوقف البرنامج عن العمل ام ماذا سيفعل الأن دعنا نقوم بأختبار المصفوفه كيف ستتعامل مع البيان الذى قمنا بادخاله

ثم قمنا بعمل كود يقوم بجمع البيانات التى قمنا بادخالها

```
Console.WriteLine(QU.Dequeue + QU.Dequeue)
```

لنقوم بتجربه المصفوفه ونلاحظ الناتج هل سيتوقف ام سيكمل وعلى اى اساس.
شاهد الناتج فى الصوره التاليه :-



الناتج من جمع 5+5 =55
هل هذا صحيح ؟ هل 5+5=55 ؟
بالطبع هو غير صحيح فى حاله الرقمى اما فى حاله النصى فهو صحيح وهذا يؤكد لك أن المصفوفه تتعامل مع البيانات بشكل نصى وهذا مانحن نريده او قمنا بتحديده فى المثال .فلماذا نحن متأكدون من ذلك ؟
لأن النوع النصى عندما يقوم بالجمع فهو يقوم بوضع الأجزاء بجوار بعضها وهذا مايطلق عليه عمليه الـconcatenation
او التسلسل وهى تحدث فى حاله النصي ولا تحدث فى الرقمي
مثال :-

```
Malek + saber = Malek saber
5+5 =55
10+10=1010
```

اما ما يحدث فى الحاله الرقميه
يطلق عليه عمليه الـPlus
او الجمع بالطريقه الحسابيه المعتاده
مثال:-

```
5+5=10
10+10=20
```


ثم قمنا بعمل مثال أخر ولكن قمنا بتحديد نوع التعامل مع الداتا داخل المصفوفه يكون رقمى فقمنا بحجز مخزن نسخه من المصفوفه كيو

```
Dim QU2 As New Queue(Of Integer)
```

على شرط ان يتعامل مع البيانات التى ستدخل للمصفوفه على انها رقميه

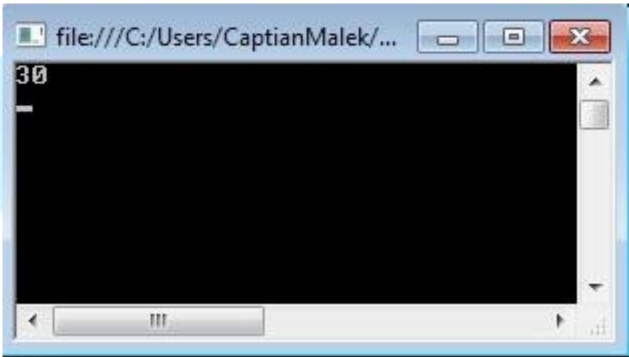
ثم قمنا باضافه بيانات رقميه داخل المصفوفه

```
QU2.Enqueue(10)
QU2.Enqueue(20)
```

ثم امرنا شاشه الكونسول ان تعرض ناتج جمع البيانات التى قمنا بادخالها فى المصفوفه

```
Console.WriteLine(QU2.Dequeue + QU2.Dequeue)
```

فكان الناتج كما فى الصوره التاليه



هذا ناتج جمع البيانين مابداخل المصفوفه 10و20

30=20+10

هذا صحيح اذا تأكدنا انها تتعامل مع البيانات على اساس انها رقميه
حتى لو قومت بوضع هذا البيان بين علامتى تنصيص مثل البيان النصى فهى ستتجاهل علامات التنصيص وتتعامل معه على اساس انه رقمى
وقم بالتجربه بنفسك ولاحظ النتائج.

انتهت الوحده العاشره :

الوحده الحاديه عشر والأخيره فى الكوليكشن :-

LIKED LIST

اللينكد ليست .. تعتبر اللينكد ليست نفس عمل الأراى ليست التى قمنا بشرحها فى الوحده الأولى ولكن الأختلاف هنا ان اللينكد ليست هى ميثود من الكلاس جنيرك الذى قمنا بشرحه فى الوحده السابقه ومعنى انها من الجنيرك اذا لابد من تحديد نوع الداتا التى سوف تتعامل معاها داخل المصفوفه كما شرحنا فى الوحده السابقه .
الأختلاف الاخر انها تحتوى على خواص جديده غير موجوده فى الأراى ليست مثل التسجيل قبل بيان معين او بعد بين معين فمثلا ان قمنا بتسجيل بيان فى البدايه وتسجيل بيان فى النهايه نستطيع تسجيل هذا البيان قبل بيان فى المنتصف أو فى الأول على حسب مانحدد له.
ايضا عندما نقوم بعرض النتائج على شاشه الكونسول نستيطع عرض عنصر قبل عنصر اخر بأسلوب اوضح نتحكم فى طريقه العرض من حيث ترتيب الأصناف التى بداخل اجزائ المصفوفه لينكد ليست . اما فى الأراى ليست لم نستطيع فعل ذلك كل ماكان علينا ان نعطيها رقم الأندكس وتعطينا البيان الخاص بهذا الرقم .. قم بمراجعه الوحده الأولى
ولكى نستطيع استخدام اللينكد ليست لابد من استيرادها من الجنيرك بنفس الطريقه التى نستورد بها الجنيرك عن طريق الكود :-

```
Imports System.Collections.Generic
```

بعض الخصائص او الميثود الخاصه بالنيم فاليو كوليكشن

الخاصيه Value
هذه الخاصيه تمكننا من معرفه قيمه بيان معين بداخل اللينكد ليست

الخاصيه Frist
هذه الخاصيه تمكننا من معرفه البيان المسجل فى الجزء الاول بداخل اللينكد ليست
ويمكننا استخدامها فى العرض على شاشه الكونسول اول بيان داخل المصفوفه عن طريق الأتى

```
Console.WriteLine(LL.Frist.Value)
```

اى قم بعرض قيمه مافى البيان الأول بداخل اللينكد الليست

الخاصيه Last
هذه الخاصيه تمكننا من معرفه البيان المسجل فى الجزء الأخير بداخل اللينكد ليست ويمكننا استخدامها فى العرض على شاشه الكونسول اخر بيان داخل المصفوفه عن طريق الأتى

```
Console.WriteLine(LL.Last.Value)
```

اى قم بعرض قيمه مافى البيان الأخير بداخل اللينكد ليست

الخاصيه AddFrist
هذه الخاصيه تمكننا من اضافه بيان داخل اللينكد ليست ويكون البيان فى اول جزء بها

الخاصيه AddLast
هذه الخاصيه تمكننا من اضافه بيان داخل اللينكد ليست ويكون البيان فى اخر جزء بها ويمكن استخدامها اكثر من مره فى المصفوفه فاذا استخدمت بطريقه تكراريه فى نفس المصفوفه فانها تقوم باضافه البيان بعد اخر بيان فيها .

الخاصيه AddAfter
هذه الخاصيه تمكننا من اضافه بيان داخل اللينكد ليست ويكون البيان بعد بيان معين نقوم بتحديده فتخيل معى انى قولت LL.AddAfter(LL.Frist,"Hello")
اى قم باضافه البيان Hello بعد البيان المسجل فى أول المصفوفه

الخاصيه AddBefore
هذه الخاصيه تمكننا من اضافه بيان داخل اللينكد ليست ويكون البيان قبل بيان معين نقوم بتحديده فتخيل معى انى قولت LL.AddBefore(LL.Last,"Hello")
اى قم باضافه البيان Hello قبل البيان المسجل فى اخر المصفوفه

الخاصيه Remove()
تمكننا هذه الخاصيه من مسح بيان معين داخل اجزاء المصفوفه لينكد ليست وذلك عن طريق كتابه اسم البيان المراد مسحه بين قوسين كالأتى :-
Remove("MyItemName")

الخاصيه RemoveFirst
تمكننا هذه الخاصيه من مسح البيان الأول فى المصفوفه لينكد ليست

الخاصيه RemoveLast
تمكننا هذه الخاصيه من مسح البيان الأخير فى المصفوفه لينكد ليست

والأن دعنا نأخذ مثال يوضح كيفيه استخدام اللينكد ليست بألاكواد
تابع معى ...

```
Imports System.Collections.Generic

Module module1
    Sub main()


        Dim S() = {"Nour", "Malek", "Shafee", "Salma"}

        Dim LL As New LinkedList(Of String)(S)

        Dim I = LL.First
        Do While I IsNot Nothing
            Console.WriteLine(I.Value)
            I = I.Next
        Loop
        Console.ReadKey()

    End Sub
End Module
```


الشرح

اولا : قمنا باستيراد دوال الـ Collections لنستطيع استعمال اللينكد ليست عن طريق الكود

```
Imports System.Collections.Generic
```

ثم قمنا بحجز مصفوفه عاديه ووضعنا بأجزائها بعض البيانات النصيه

```
Dim S() = {"Nour", "Malek", "Shafee", "Salma"}
```

ثم قمنا بحجز مصفوفه من النوع لينكد ليست على شرط انها تتعامل مع البيانات من النوع النصى فقط .. ثم امرناها ان تأخذ محتويات المصفوفه العاديه

```
Dim LL As New LinkedList(Of String)(S)
```

اعلم انه لابد من تحديد نوع البيانات التى لابد ان تتعامل معها المصفوفه لينكد ليست لأنك لو لم تقم بتحديدها لن تعمل معك .

ومعنى اننا أمرناها ان تأخذ محتويات المصفوفه العاديه .. هذا يعنى وكأنك قمت بعمل اضافه داخل المصفوفه لينكد ليست واضفت بها نفس الاجزاء التى قمت باضافتها فى المصفوفه العاديه

ثم قمنا بعمل دواره تقوم بعرض محتويات المصفوفه لينكد ليست

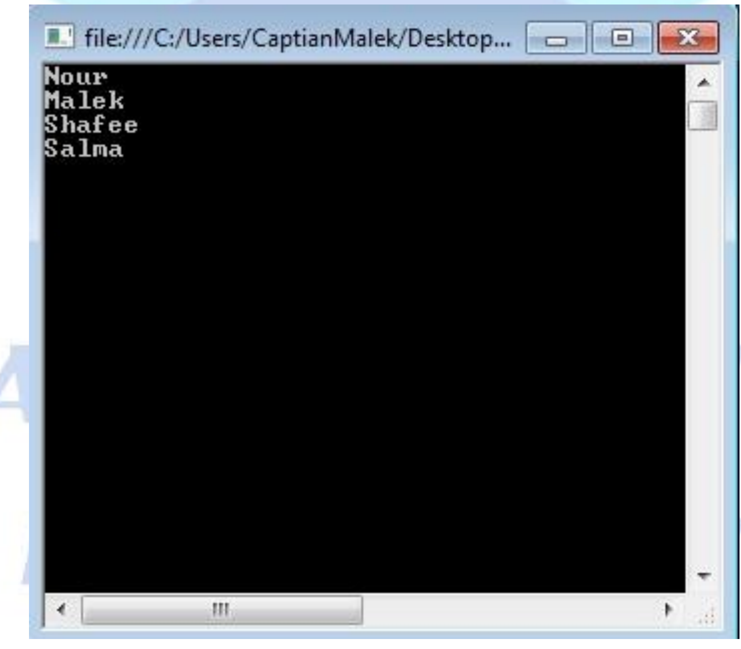
```
Dim I = LL.First
Do While I IsNot Nothing
Console.WriteLine(I.Value)
I = I.Next
Loop
```

قمنا بحجز متغير ووضعنا بها اول بيان فى المصفوفه لينكد ليست

ثم قمنا بعمل دواره .. تدول طالما المتغير غير خالى (اى محتوى على قيم من اللينكد ليست)

تدور الدواره وفى كل لفه تقوم بعرض القيم التى بداخل المتغير حتى يصل المتغير الى اخر عنصر فى المصفوفه فينتهى عمل الدواره بعرض كل محتويات الـلينكد لـيست

شاهد النتيجه فى الصوره التاليه :-



انتهت الوحده الحاديه عشر والاخيره فى الكوليكشن :

الوحده الثانيه عشر والأخيره فى هذا الجزء:-

COMPREESION (ZIP) & DECOMPREESION

الضغط بالصيغه Zip وفك الضغط عن الملفات عن طريق الكود
ستعلم فى هذه الوحده كيف نقوم بضغط الملفات وفك الضغط عنها وذلك بأستخدام كلاس
بداخل مكتبات الانبوت اوت بوت InPutOutPut
التى قمنا بالحديث عنها فى الجزء الأول من هذا الكتاب.
هذا الكلاس يسمىCompreesion
اذا لابد قبل استعمالها استيرادها فوق الموديول او مايسمى بمنطقه التصريحات العامه عن طريق الكود التالى :-

```
Imports System.IO.Compression
```

وايضا سوف نقوم باستخدام المكتبه الأساسيه InputOutPut
لكى نستطيع انشاء ملف بالكود لنقوم بضغطه والتجربه عليه وذلك عن طريق الكود التالى:-

```
Imports System.IO
```


ملحوظه!!
ان كنت لاتتذكر كيف يتم العمل على هذه المكتبات من فضلك راجع الجزء الأول من هذا الكتاب فهو يتكلم بالتفصيل عن هذه المكتبات وكيفيه انشاء فولدرات وملفات والكتابه بداخل الملفات ومسح ونقل الملفات من اماكن معينه الى اماكن اخرى .... الخ .

والأن دعنا نبدأ فى كتابه الأكواد تابع معى المثال التالى لانشاء الملفات وضغطها:-



```
Imports System.IO.Compression
Imports System.IO

Module module1

    Sub main()


Dim GZIP As New GZipStream(File.Create("C:\Test.Zip"),
CompressionMode.Compress)

        Dim SW As New StreamWriter(GZIP)

        For i As Double = 0 To 100
            SW.WriteLine("Hello")
        Next

        SW.Close()
        GZIP.Close()

        Console.WriteLine("Done")
        Console.WriteLine("Press Enter If You Like To Decomprees This File")
        Console.ReadKey()



 ' لفك ضغط الملف
Dim Zip As New GZipStream(File.OpenRead("C:\Test.Zip"),
CompressionMode.Decompress)

        Dim SR As New StreamReader(Zip)
        Console.WriteLine(SR.ReadToEnd)
        Console.ReadKey()

        Zip.Close()
        SR.Close()

    End Sub
End Module
```


الشرح

اولا : قمنا باستيراد دوال الـ InputOutPut لنستطيع استعمال الكومبريشن عن طريق الكود

```
Imports System.IO.Compression
```


اولا : قمنا باستيراد الـ InputOutPut لنستطيع استعمالها فى انشاء الملفات عن طريق الكود

```
Imports System.IO
```

ثم قمنا بحجز متغير من النوع GZipstream الذى يتعامل مع الملفات المضغوطه

```
Dim GZIP As New GZipStream(File.Create("C:\Test.Zip"),
CompressionMode.Compress)
```

وقولنا له قم بأنشاء ملف مضغوط على السى بالأسم والامتداد Test.Zip

لاحظ ذلك الكود CompressionMode.Compress

هذا يعنى اي اجعله على مود او نمط الضغط ولابد ان نحدد له المود او النمط الذى سيعمل عليه ويمكننا ان نجعله على مود او نمط فك الضغط .

ثم قمنا بحجز مخزن من النوع ستريم رايتر الذى يقوم بالكتابه داخل الملفات كما شرحنا فى الجزء الأول من هذا الكتاب وامرناه ان يكتب داخل الملف المضغوط الذى قمنا بانشاءه مسبقا

```
Dim SW As New StreamWriter(GZIP)
```

ثم قمنا بعمل دواره بسيطه تقوم بالدوران مائه مره وفى كل لفه تقوم بكتابه كلمه Hello داخل الملف المضغوط

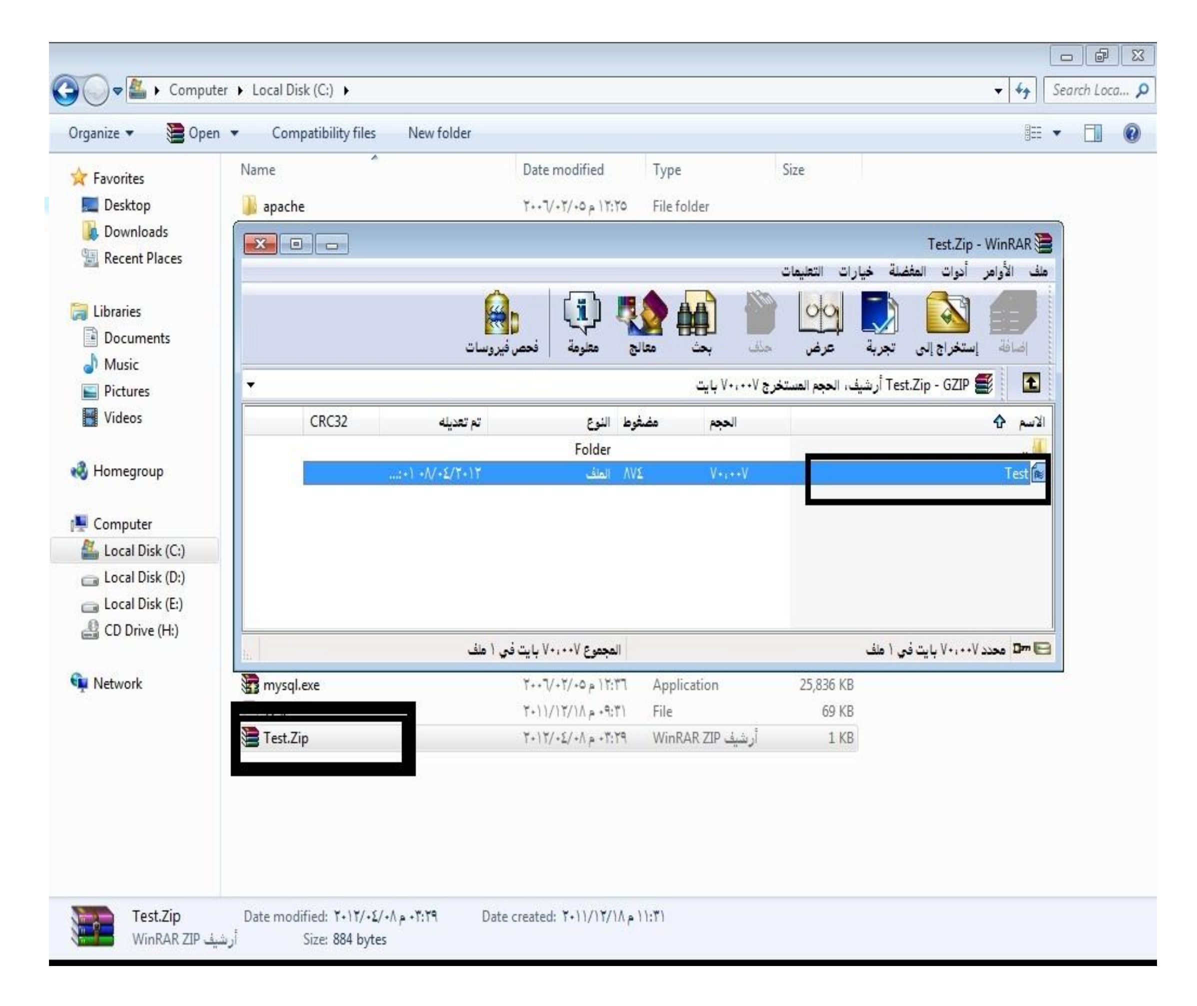
```
For i As Double = 0 To 100
SW.WriteLine("Hello")
Next
```

بعدما ينتهى عمل هذا الدواره يصبح الملف بداخله 100سطر بكل سطر كلمه Hello

ثم قمنا باغلاق المخزن Gzip , sw كى لا يكون حملا على الميمورى

```
SW.Close()
GZIP.Close()
```

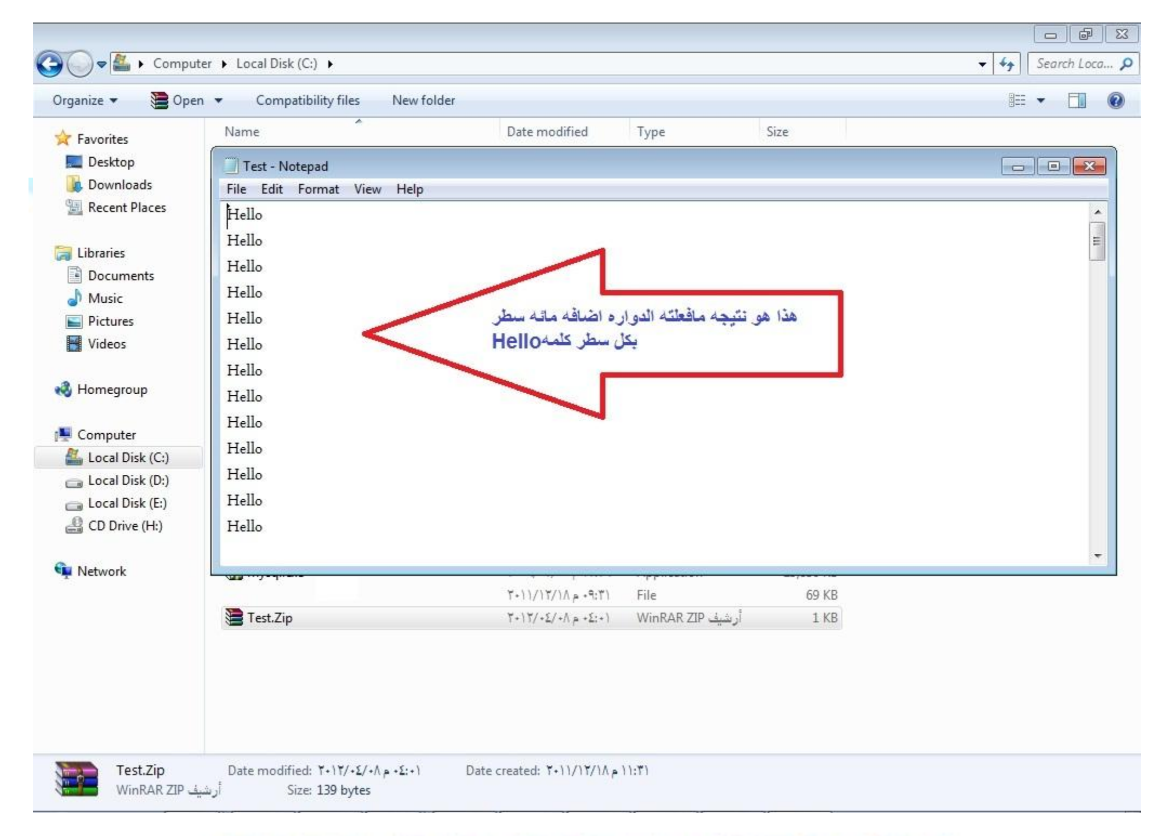
دعنا نذهب الى المسار الأن ونرى هل نشىء الملف فعلا ام لا
شاهد النتيجه فى الصوره التاليه :-



كما ترون لقد تم انشاء الملف المضغوط وانشاء ملف بداخله والكتابه عليه
اذا دعنا نرى ماماقومنا بكتابته داخل هذا الملف
قم بفتحه عبر الـ NotePad وشاهد النتيجه .

شاهد النتيجه فى الصوره التاليه :-





والأن دعنا نكمل باقى المثال لنشرح كيف نقوم بفك الضغط عن الملف بالكود .

ثم قمنا بحجز متغير من النوع GZipstream الذى يتعامل مع الملفات المضغوطه

```
Dim Zip As New GZipStream(File.OpenRead("C:\Test.Zip"), CompressionMode.Decompress)
```

وقولنا له قم بفك الضغط عن الملف المضغوط على السى بالأسم والامتداد Test.Zip

لاحظ ذلك الكود `CompressionMode.Decompress`

هذا يعنى اي اجعله على مود او نمط فك الضغط ولابد ان نحدد له المود او النمط الذى سيعمل عليه كما قولنا مسبقا.

ثم قمنا بحجز مخزن من النوع ستريم ريدر الذى يقوم بالقراءه من الملفات كما شرحنا فى الجزء الأول من هذا الكتاب وامرناه ان يتعامل مع الملف المضغوط الذى قمنا بانشاءه مسبقا

```
Dim SR As New StreamReader(Zip)
```

اذا هو جاهز الأن ان يقرأ مابداخل هذا الملف المضغوط

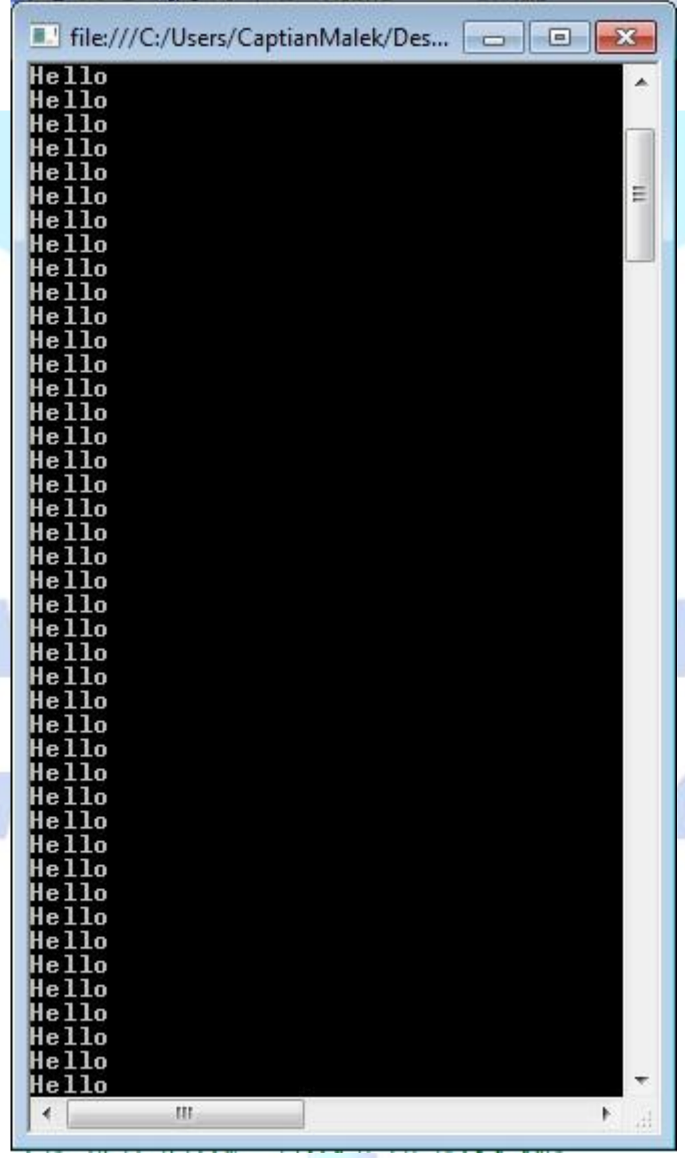
```
Console.WriteLine(SR.ReadToEnd)
```

ثم امرناه ان يقرأ مابداخل هذا الملف حتى النهايه ثم يقوم بعرضه على شاشه الكونسول بدون الحاجه لعمل دواره سيقوم بقراءه كل مافى هذا الملف حتى النهايه

وان كنت لاتفهم كيفيه عمل ستريم ريدر قم بمراجعته فى الجزء الأول من هذا الكتاب

بعدما ينتهى من القراءه سيعرض الكونسول 100سطر بكل سطر كلمه Hello

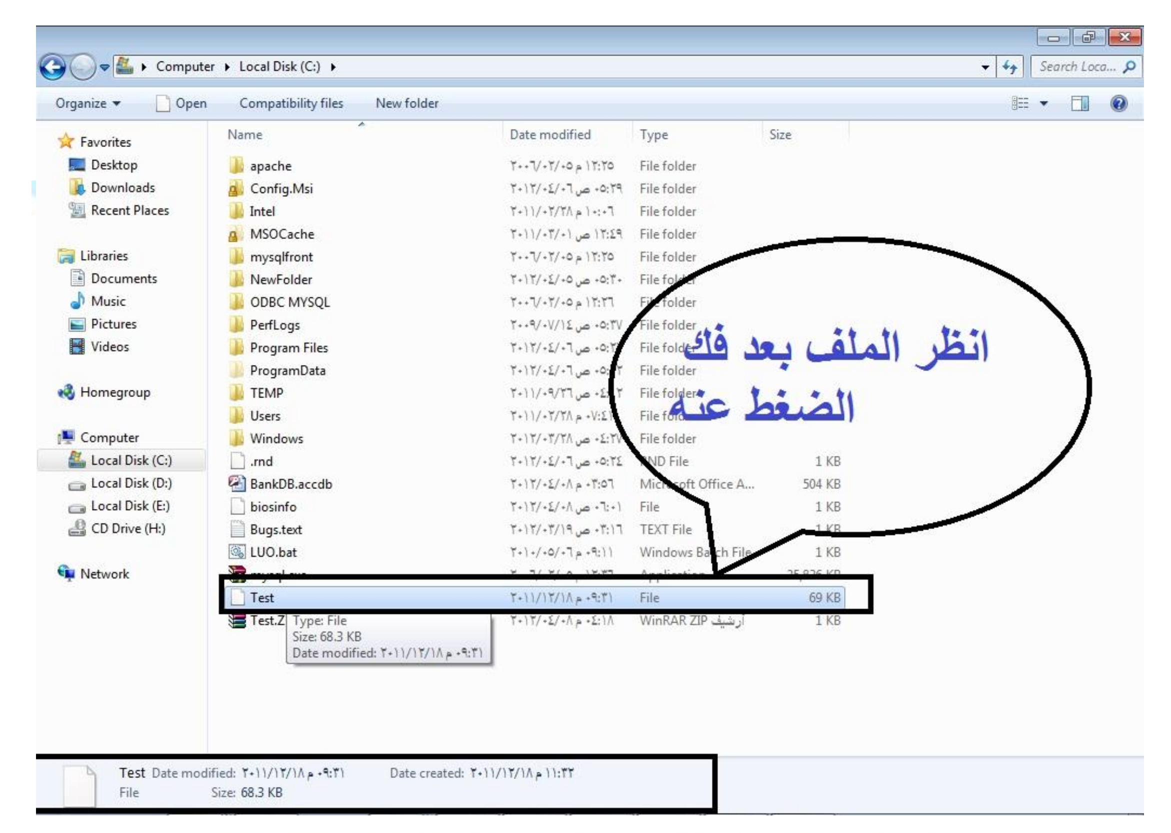
تابع النتيجه معى فى الصوره التاليه :-



ثم قمنا باغلاق المخزن SR, Zip كى لا يكون حملا على الميمورى

```
Zip.Close()
SR.Close()
```


والأنا ذا ذهبنا الى المسار C:\
سنجد ان الملف بالفعل فك ضغطه .. كما ترون فى الصوره التاليه:-



انتهت الوحده الثانيه عشر :



انتهى الجزء الثانى من الكتاب

EASY BOOK IN FRAMEWORK

تم الاصدار فى 2012/4/15

اسأل الله ان تكونو قد استفدتم بما فيه من محتويات ..
هكذا نكون قد انتهينا بشرح الفريم ورك ويعتبر هذا الجزء الأخير فى شرح الفريم ورك
وفى الجزء القادم ان شاء الله سنقوم بشرح الطريقه المبتكره لهندسه صناعه البرمجيات
الا وهى (Opject Oriented Programing (OOP
وسنقوم بتطبيق مشروع بنك الـ Hsbc داخل لغه البرمجه VB.Net
بالـ Windows Forms

المشاريع المطلوبه فى هذا الجزء:-

اولا :-

تطبيق كل الأمثله التى قمنا بشرحها يدويا وأرجو ان تنسخ الاكواد .. بل قم بكتابتها بيدك لكى تتعود على طريقه كتابه الأكواد ولكى يصبح ذهنك مستعد لطريق البرمجه وسوق التكنولجيا

ثانيا :-

يقوم كل طالب بتطبيق فكره مشروع بناءا على ماقمنا بدراسته ويكون هذا المشروع يجمع اكبر كم من الأكواد التى قمنا بدراستها .. وكلما كان المشروع يجمع اكبر عدد من الأكواد والخواص زاد التقدير لهذا المشروع

ايضا ان خطر ببالك فكره اى مشروع اخرى او اعاقه فى تطبيق مثال او كود معين او ان كنت تريد تقييم مشروعك قم بمراسلتنا عبر الايميل التالى :-

Captian.Malek@Gmail.com

اعلم اخى القارىء

انك اذا قمت بتطبيق المشاريع المطلوبه منك اعلاه
انك فهمت الكورس فهما جيدا ولا تيأس ان حدث معك اى خطأ فى تطبيق مثال معين مهما كانت جسامة هذا الخطأ فان اكثر الخطائون المتعلمون .. فكلما زادت اخطاءك اعلم بأنك على سلم النجاح .. ولا تيأس يوما ما الى ان تصل الى الدرجه الأخيره من هذا السلم

فى انتظار تقيماتكم للكتاب عبر الايميل

مع تمنايتى لكم بالتوفيق والنجاح

قام بكتابه وتأليف الكتاب
المهندس/مالك صابر محمد

قام بالمراجعه والتصحيح
الاستاذ/ عبدالله صابر محمد